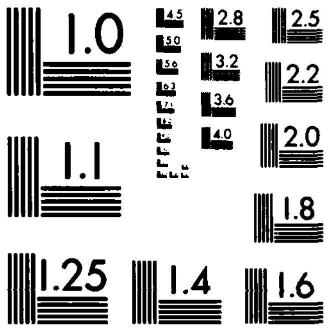
AD-A132 899 DEPARTMENT OF DEFENSE COMPUTER TECHNOLOGY A REPORT TO CONGRESS(U) OFFICE OF THE UNDER SECRETARY OF DEFENSE FOR RESEARCH AND ENGINEERING WASHINGTON DC AUG 83 1/2

UNCLASSIFIED F/G 9/2 NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

3

AD-A132899

DEPARTMENT OF DEFENSE COMPUTER TECHNOLOGY

A REPORT TO CONGRESS

AUGUST 1983

OFFICE OF THE UNDER SECRETARY OF DEFENSE
RESEARCH AND ENGINEERING
WASHINGTON, D.C.

DEPARTMENT OF DEFENSE

COMPUTER TECHNOLOGY

A REPORT TO CONGRESS

August, 1983

OFFICE OF THE UNDER SECRETARY OF DEFENSE

RESEARCH AND ENGINEERING

WASHINGTON, D.C.

TABLE OF CONTENTS

TABLES

FIGURES

FOREWORD

This report contains the results of an internal DoD study of mission-critical computer technology ~~directed by~~ Congress ~~in~~ the Fiscal Year 1983 Defense Authorization Act.

Further direction is included in the report of the Joint Conference on the FY1984 authorization. Where possible, this additional direction was taken into account. However, detailed cost and schedule information are not available for inclusions in this report. They will be provided separately.

It is suggested that the reader begin with The Executive Summary and Chapter 6, "Plan for Proliferation Reduction". The former summarizes the Congressional direction, sketches pertinent background and presents the conclusions and recommendations of the study group.

The recommendations provide a strategy to evaluate prospects for a new generation of computer technology while precluding a dislocation in capability should such not materialize or should it be delayed. A joint Service program to lead to more relaxed standards, relying more on interfaces, is proposed. This program has not been fully agreed to by the Services nor has it been detailed with respect to cost and schedule. The Navy has programmed 6.3 funds at a modest level beginning in FY85 and intends to reprogram FY84 funds to support preparatory work. We anticipate that the Army and Air Force will be able to similarly reprogram FY84 and FY85 funds or redirect ongoing efforts to make this a joint Service effort.

The remainder of this report is dedicated to discussion of the needs for non-commercial computers for military applications as well as delineation of areas where needs can be satisfied from the commercial sector.

EXECUTIVE SUMMARY

This Report presents the results of a study conducted by the Department of Defense (DoD) at the direction of Congress in their Conference Report accompanying S.2248, "Department of Defense Authorization Act, 1983." The study was commissioned to address the following specific issues:

(1) The applicability of commercial computer technology (including, but not limited to computer hardware, semiconductors and associated Instruction Set Architectures (ISAs)) to Defense Department missions.

(2) The desirability of standardizing at the ISA level in light of alternative approaches.

(3) The degree of software transportability (transferring computer programs from one computer to another) that the various approaches permit and how each approach would affect Department of Defense hardware/software logistics and costs of ownership.

(4) The relative merits and liabilities involved in the incorporation of each approach into Department of Defense weapons systems.

(5) Justification for all on-going Service computer development projects.

(6) A plan to reduce the proliferation of these computers.

Background

Congress' interest in these issues arose as the Department proposed a policy to reduce unnecessary proliferation of militarized computers and to facilitate both the software development process and the reuse of software across systems. This proposed policy has two major facets: The first is to standardize on a modern high-order computer programming language, Ada[1]. The second is to manage the interface between the resulting software and the target military computers upon which it must execute.

Defense system software, particularly that which has come to be termed "mission-critical[2]" is traditionally beset with cost and schedule problems. Among them are included:

1. Ada is a Registered Trademark of the U.S. Department of Defense.
2. "Mission-critical" refers to those applications involving: (1) intelligence systems; (2) cryptography for National defense; (3) command and control of military forces; (4) weapons or weapons systems; and (5) direct support to military or intelligence operations.

(1) System schedule slip due to software delays.

(2) System failures due to software errors.

(3) Soaring cost of software.

(4) Inability to reuse and transport applications and support software.

(5) High cost of logistics and maintenance support.

(6) Low operational availability (survivability, failure and casualty backup, continuity of operations).

(7) Difficulty in maintaining and upgrading software as needs change due to a dynamic threat and hardware evolution, and as the user gains experience with the system and evolves his employment tactics.

(8) Lack of adequate competition.

Conclusions

Based on both software and hardware arguments, it is concluded that object code transportability, hence ISA standardization, is essential to improve competition, to reduce the costs associated with the computer life-cycle and to provide for acceptable survivability and continuity of operations of combat systems. Moreover, some mechanism for object code compatibility will remain essential until improvements in the state-of-the-art permit otherwise. The programs established in each Service to satisfy growing mission-critical computer needs should not be interrupted now in an attempt to achieve a greater rate of convergence.

ISA management is, at current state-of-the-art and practice, a valuable approach to provide object-code transportability. It is not without trade-offs, but there is no evidence that it is, per force, inappropriate. To the contrary: The ISA interface is the minimum level of management which can maintain vendor-independence, does not dictate detailed design parameters for either hardware or software and which can facilitate transportability and reuse of time-independent software across differing hardware implementations or realizations.

There is a level between ISA and HOL which may be valuable in some cases. This is the run-time operating system (RTOS) which could provide a "software-only" interface to applications programs and a second "ISA software" interface to the hardware. Not all applications fit this "system architecture" model, but where it is appropriate it may provide a satisfactory compromise, assuming that there is not proprietary bundling involved. Although this approach was not studied, per se, it obviously will not alleviate proliferation problems and, without ISA management, might actually aggravate them.

The CODSIA team which is studying this issue from the industry's standpoint has developed a hybrid approach which they term "Form-Fit-Interoperability-Transportability" or FFIT. Their report is not yet available for review.

Neither RTOS nor FFIT solve the hardware-related issues but they will provide an improvement on HOL standardization alone.

Recommendations

Section 6 contains a preliminary strategy for reduction of unproductive proliferation of military computers. It is necessary to emphasize that this is not intended to apply to use of common commercial items but only to those to which support "mission-critical" applications. The approach is intended to provide a balance among:

- o Competition
- o Technology insertion, and
- o Proliferation reduction.

Recommendations are offered in both the non-mission critical and the mission-critical application areas:

(1) Non-Mission-Critical Computers and Systems

(a) Ada should be a candidate language, allowed to compete on its merits and taking into consideration the availability of development and support environment. If Ada is used, then delivered software must be processed by a validated compiler.

(b) There should be no policy-level control of ISAs or of hardware design. Interface and protocol standards may be applied to support system integrity.

(c) Normal requirements and procurement processes should be applied.

(d) Common commercial products, both hardware and software, should be used wherever they are adequate to fulfill missions needs in accordance with DoD Directive 5000.37, "Acquisition and Distribution of Commercial Products (ADCP)".

(2) Mission-Critical Computers and Systems

(a) Use of Ada should be mandatory with necessary exceptions to be sought and approved by a formal waiver process on a case-by-case basis. (Services have been directed to submit Ada Introduction Plan for review and approval of DUSDRE).

(b) Development of the Ada Language System (ALS) and the Ada Integrated Environment (AIE) should be continued. The Army and Air Force should review the practicality of merging these efforts or of choosing between them for joint development and use. The System Interface Standards (SIS) being jointly developed should, in either case, be rigorously applied. The Navy should proceed to build upon ALS by adding specific tools and target generators to yield the Navy Ada Language System, ALS/N, which should be afforded joint configuration management with ALS and should rigorously implement SIS. Regulations should be reviewed and revised as necessary to assure no hinderance to providing DoD-developed Programming Support

Environments (PSEs) to Industry for use on DoD programs. It should be recognized that PSEs must be improved and extended over time but that configuraton management must be maintained in an upward-compatible manner. The Ada Joint Program Office (AJPO) should closely monitor commercial APSE developments and serve as a central source of information for the Services.

(c) Jointly with Industry, the Services should assure adequate testing of PSEs before making their use mandatory for processing delivered software. There should be an assured capability to incorporate private-sector tools while protecting proprietary items.

(d) The Department should maintain an interim policy of object-code compatability to bound unnecessary proliferation of both hardware and software until a viable alternative is demonstrated.

(e) Services should jointly investigate potential alternative approaches such as language machines, reduced instruction set computer (RISC), non-von Neumann architectures, run-time interface standards, etc. The objective is to perform a comprehensive/requirements/opportunities evaluation as was done for Ada (See Section 5).

(f) The Army and Air Force should join the Navy as full partners in the effort to provide a standardized, high-technology, embeddable microprocessor for tri-Service use. Use of VHSIC and VHSIC-like technology should have top priority. MIL-STD-1750 and MIL-STD-1862 should be considered candidates along with commercially-developed architectures.

(g) For non-Mil-Spec mission-critical environments, Services should be encouraged to use either common-commercial, ruggedized-commercial or "off-the-shelf" militarized computers based upon the performance requirements of the specific application. Full consideration should be given to Ada-based systems where there is no strict hardware interchangeability requirement. By "Ada-based" is meant an Ada high-level language machine or a combination of a full, validated Ada PSE and associated computer of choice (not necessarily a computer of any specific "object code" features).

Further Background

The following paragraphs expand upon the background for language standardization and ISA management and identify other possible alternatives to reduce proliferation and contain costs. These topics are treated more fully in the body of the report:

Language Standardization:

In the mid-1970s, there was specific attention from the Deputy Secretary of Defense to the issue of computer resource management. The first DoD policy was issued in 1976 as DoD Directive 5000.29, "Management of Computer Resources in Major Defense Systems". It directed, inter alia, that software be developed in a high-order language, or HOL. The purpose was to facilitate application of more modern software engineering practices and to reduce dependence on specific, proprietary software and computers. Prior practice had led to inefficiency and dependence on incumbent suppliers, resulting in restricted competition — often sole-source.

Where actually used, HOL programming proved valuable in improving quality and schedule performance, and hence, helped reduce costs. However, the issues surrounding availability of given HOLs for various military computers and associated software development environments, as well as the capabilities of available languages, served to attenuate policy. Concern over lack of an adequate language for DoD-wide application generated the DoD Common Language Program. This tri-Service effort led to the development of the language Ada.

Ada is the first computer programming language to be developed from a comprehensive, user-generated set of requirements. It was intended to span the mission-critical needs of DoD, particularly those evolving from real-time applications. At the current state-of-the-art and practice, however, Ada (or any other language) does not fully isolate the user and the applications software from details of the hardware environment upon which software must execute for two salient reasons:

First, direct use of the ISA (rather than the HOL) is required to develop certain portions of the software such as:

(1) parts of the run-time operating system,

(2) input-output programs,

(3) time-critical functions for which the program produced by the HOL compiler does not meet the required system response time,

(4) functions for which the program produced by the HOL compiler requires large amounts of memory such that the computer's memory capacity is exceeded, and

(5) functions for which features are not available in the HOL of choice.

Second, HOL programs themselves are also ISA-dependent. Dependence of HOL software on the ISA for which it was developed exists with respect to all high order languages (including Ada) used to develop systems. A program can produce radically different results (computations) in systems with different ISAs.

Ada has been designed with the goal of minimizing ISA dependencies. With the current state-of-the-art limitations, however, dependencies remain, as discussed in Section 4.4.2.

Thus, significant and critical portions of both embedded software and support software can be expected to have to be modified or redesigned if a given target computer were to be replaced with one having a different ISA. ISA dependencies are pervasive within high-order languages, within the programming support environment (PSE) and in run-time software (such as a real-time operating system). ISA dependencies are unavoidable in current practice and reflect the state-of-the-art in both software and hardware design. These dependencies are an impediment to reuse and transportability of software and hence to maximum utilization of rapidly evolving hardware technology.

It is apparent, then, that standardization at the programming language alone, although it provides a significant improvement over no policy, is not a complete solution to the problems associated with proliferation. Clearly, it does nothing for hardware logistics and only little for the enhancement of competition.

ISA Management:

The concept and rationale for ISA management involve two principal issues: The reuse (or transportability) of both development (support) software and applications software across mulitple installations and systems. A common interface between hardware and software provides a viable basis for open, unencumbered competition for both software and hardware products.

The first activity toward mangement of the software/hardware interface was the drafting of proposed DoD Instruction 5000.xx, "Interim List of Approved Computer Architectures." This proposed policy, patterned after earlier work on languages, was:

> "...designed for use in conjunction with (DoD) Directive 5000.29 and DoD Instruction 5000.31 "Interim List of DoD Approved High-Order Programming Languages (HOL)") to reduce the proliferation of computer [Instruction Set] Architecture as incorporated throughout defense systems and subsystems requiring a military environment; to limit and stabilize the combinations of computer architecture and HOLs which must be supported in software (without sacrifice to technological innovation and vendor competition); and to ensure the control of those computer architectures which are approved."

The interim list contained six instruction set architectures (ISAs) as implemented in nomenclatured hardware in wide use in the Services at that time. This was an interim list; it was intended to include more advanced vendor-independent ISAs over time and to remove those which proved to be technically or economically inappropriate. By including plans to update the list at least every two years, any trend toward technical or operational obsolescence could be quelled. The Proposed Instruction met resistance from some who felt DoD's intention was to standardize hardware, per se, and those who felt that, since some of the ISAs were derived from proprietary designs, incumbent vendors would be favored.

A study panel was formed to review the issues and to recommend whether such as Instruction was appropriate and, if so what form it should take. The Panel reported on March 20, 1980. They recommended that: "DoD issue an Instruction ... limiting the number of ISAs to be used in embedded systems." The rationale for the recommendation was that it would reduce overall software cost through reuse of satisfactory existing software, transportability of runtime software in critical military situations and avoidance of costs associated with training and the time to redesign serviceable software for new ISAs. A copy of the Executive Summary from the Panel's report is included as Appendix I. The Panel's redraft of 5000.xx became Proposed DoD Instructon 5000.5x.

The issue was further studied by a special Task Force of the Defense Science Board. The Task Force concluded that the controversy over ISA configuration: "... is largely between interests which have as primary concerns the operation, maintenance, deployment and post-acquisition support of these same computers." The Task Force recommended that 5000.5X be issued after review for consistency with acquisition regulations and clarification of scope.

There remained, in some quarters, concern over potential obsolescence, if DoD were to "standardize" the software/hardware interface. It is well here to recall the works of Fred Brooks[4]:

"The last woe, and sometimes the last straw, is that the product over which one has labored so long appears to be obsolete upon (or before) completion. Already colleagues and competitors are in hot pursuit of new and better ideas. Already the displacement of one's thought-child is not only conceived, but scheduled.

"This always seems worse than it really is. The new and better product is generally not available when one completes his own; it is only talked about. It, too, will require months of development. The real tiger is never a match for the paper one, unless actual use is wanted. Then the virtues of reality have a satisfaction all their own."

Admiral Gorshkov put it more succinctly:

"The best is enemy of the good enough."

Thus the objective of ISA management is to deal with the dynamic balance among "what is", "what is to be", and "what should no longer be".

Additional Standardization Approaches:

Additional measures which have been studied in detail include:

(1) "Total Product Standardization," as exemplified first in the Navy's AN/UYK-7 and AN/UYK-20 computers, involves production of single design for each computer type. The Military Computer Family (MCF) AN/UYK-41/49 and the AN/UYK-43/44 are latter-day examples of this approach.

(2) "Form-Fit-Functon (F^3) of Computer Modules" involves computers containing modules where each module type would be available from several suppliers. Each supplier's design would be different yet all modules of a given type would be interchangeable.

(3) Form-Fit-Function at the "Box-Level" involves specifications only with respect to the external attributes of each type of computer. Each type of computer would be available from several suppliers and

4. Brooks, Frederick P., "The Mythical Man-month": Essays on Software Engineering;" Addison-Wesley; Reading MA; 1975.

each supplier's design would be different. All computers of a given type would meet the same functional and performance specifications to facilitate a system design that would accommodate interchangeability in the field.

(4) "Limited External Attributes" would allow, for example, software and interface compatibility. This approach places no constraints on (internal) hardware design and thus, systems designed to this approach have little to offer with regard to field interchangeability, logistics support, etc.

(5) "ISA-only" means that computers would be common only at the level of the software/hardware interface.

SECTION 1 OUTLINE OF PROBLEM AND STUDY APPROACH

1.1 Tasking

The United States Congress is paying increasing attention to the policies and costs associated with the accelerated emphasis on computerized defense systems. Concern has arisen over the Department of Defense's (DoD's) attempt to standardize the computer hardware/software interface.

Draft DoD Instruction 5000.5x was proposed to identify a set of standards for computer Instruction Set Architectures (ISAs). An ISA is the specification of the interface between software and hardware. It includes the attributes of a computer as may be seen by a machine [assembly] language programmer or the target code generator of a compiler for a high-order language (HOL). It describes the conceptual structure and functional behavior of a computer as distinct from the organization of the data flow and controls, logic design or physical implementation.

In order for software programs to be written, it is necessary to establish the language desired (or required) and the instruction set architecture of the machine on which the software is to operate. The corollary of this is very important: Given the programming language and the instruction set architecture, the computer software can be written. To the person who is going to use a computer, this corollary has a number of important applications:

i. Since so many advances are being made in the hardware technology realm, it is possible to write the software first IF the instruction set architecture is available. The hardware decision can be delayed.
ii. If the same language and architecture are maintained, the investment in software that has been bought, tested and proved reliable can be maintained as hardware evolves to accommodate technology movement.
iii. Application software written by others may be reused, if the same architecture and language are maintained.

Serious efforts were made in the Seventies and continue in the Eighties to counteract the inability of users to reuse software except in a particular architecture. The Ada programming language is a major step in reducing the number of dependencies that software programs have with particular ISAs.

1.2 Historical Background

The Sixties and Seventies saw a vast growth in the use of computers within the DoD. Since World War II, the computer has, for the most part, been a resource of the strategic domain in the operational world: cryptology, intelligence, and command and control. Sputnik accelerated missile developments which, because its high-speed and unearthly trajectory (let alone the fact that it was unmanned), demanded improved computer technology for its navigational and servocontrol systems. However, the computer generally remained outside the "operational" environment of the military until the Vietnam War.

In Vietnam, the military computer came of age. As always, the impetus came from two sources: The battlefield environment into which the American Soldier was thrust (such as Surface to Air Missile (SAM) environment) which forced automated countermeasures on our part, and technological advancements (such as the smart bomb), which inspired new tactics.

But Vietnam's computerized systems--in calm, objective, postwar assessment, possessed certain characteristics which did not permit reasonable long-term management. The software subsystem was extremely expensive to develop. Worse, development seemed never to be complete, making life-cycle costs significantly more expensive than development costs. The software development process was interminably long, and, since it was invariably on the full system's critical development path, system development and fielding were pushed that much farther downstream. Therefore, software subsystems did not receive rigorous quality acceptance testing because of the unrealistic testing requirements associated with cost and schedule constraints.

After fielding, the software subsystems demanded post-deployment support over the entire system life. These costs generally run 10% to 20% of software acquisition costs per year. With most weapon systems having a system life of 15-25 years, life-cycle costs easily exceeded acquisition costs. Yet, they were rarely factored into the acquisition decision. More recent data indicate that this annual support requirement may be more nearly 30%-40% of the software acquisition cost.

1.3 Statement of the Problem

There is a basic set of problems attendant to the use of computers by DoD in strategic and tactical systems. This Section describes those problems.

1.3.1 Expensive and Lengthy System Development and Evolution

Most new automated strategic and tactical systems are for use against specific threats for which systems have never before been developed. The system may include many other technologies which are then integrated into one common weapon. The first problem is to provide an adequate definition of the scope of the system and the specifications of the system requirements. Normally there is a concept phase and then an advanced development phase in which various means of meeting the threat are analyzed and tested. This may include several parallel developments with tests to identify the best alternatives for meeting the system requirements. Depending on the type of system, simulation and modeling may be necessary because of the expense of adequate testing. For instance, flying enough aircraft against an air defense system at different altitudes and densities is very expensive. It may take several years to integrate enough equipment into a prototype system for testing. Many platforms, such as aircraft, ships, tanks or satellites are years in the making. Few, if any large systems are developed, tested, produced and fielded in less than seven years. The lengthy development may actually span a generation of computer hardware at the rate large-scale integrated circuit technology is changing.

Once the decision is made to produce a particular system, it is several years before the first production unit rolls off the production line and new equipment training teams introduce the equipment into units of the Services.

DoD has learned that using unproven technologies may further increase the schedule risk, particularly in computer hardware and software technology. These technologies should be developed off-line to the normal system development and proven before committing large expensive systems to schedules which then slip and incur large overruns. Once fielded, the operation and support phase is significantly longer than the development phase.

1.3.2 High Cost of Computer System Software

For most management information systems (MIS) common to systems used in the civilian sector, DoD develops only special applications software using "already developed" commercial operating systems, data-base management systems, etc. For most strategic and tactical systems, those products do not apply, and system-specific software, (such as operational flight programs or radar control programs) must be developed. Software of this type may be several times more expensive to develop than commercial applications software. Therefore, use of previously developed software, where possible, is important.

1.3.3 Inability to Reuse/Transport Applications and Support Software

Strategic and tactical systems are normally in application areas not supported in the civilian sector and are near-real-time, interrupt-driven systems. In the past, much of the software has been programmed in assembly language to get the necessary performance to meet the system requirements.

Assembly language programs are dependent upon tuned to the ISA being used. Consequently, at the end of the system development phase, the acquisition of a computer with the same commercial ISA updated in hardware technology is usually sole-source, since the ISA is protected and normally all MIL-SPEC computers of a particular ISA are only made by one company. If competition is held, then the chance of carrying over the already-developed software (ISA-specific) is lost if the target ISA turns out to be different. If a computer of a new ISA is being developed for a particular system, then normally the support software lags and the system is delayed. Computers of upwardly-compatible ISAs can re-use systems and support software until newer software is ready.

By the most-used rule of thumb, the software development cycle is broken down as follows:

40% for design,
20% for coding, and
40% for integration, debugging and testing.

For most systems, then, the majority of the cost for software is other than coding. Re-use of existing software--both operational and support--could be a major contributor to reducing schedules of system upgrades by reducing time for design and test. Improvements then would be primarily for necessary changes to meet new threats. It should be mentioned that Ada helps not only in coding, but the Ada technology applies to design, testing and "maintenance" to reduce costs there, as well.

1.3.4 High Cost of Logistics Maintenance and Support

Since most systems remain in the inventory for 20 years or more, maintenance of these systems during the operations and support phase more than equals the original development and acquisition cost. More than likely, the electronics will have to be upgraded during the maintenance phase, because the micro-electronics go out of production. Operational software may be supported and improved gradually during this phase maintenance and support cost over life cost for the hardware may be reduced by taking advantage of increased reliability and improved built-in test features with new hardware if the ISA remains upwardly compatible between computer hardware changes.

1.3.5 Inability to Acquire, Train and Retain Competent Military Field Maintenance Personnel

DoD is experiencing a lack of competent field maintenance personnel. Unnecessary proliferation of computer types exacerbates the need for training and retention of qualified field maintenance personnel. To the extent that maintenance training courses can be reduced, the likelihood of having trained personnel to make field repairs to computer systems is greatly increased. The direction of developments is to include built-in test circuitry to diagnose and trouble-shoot systems automatically. This trend keeps hardware costs high by including more fault-tolerant design features to improve reliability and ability of operators to maintain and repair systems on the spot.

1.3.6 Low Operational Availability

Operational availability is the measure of a complete system to be functional for a specified period. It is directly related to the state of readiness of a particular unit to perform its designated mission in combat. Combat-readiness of a unit is a function of adequate personnel being available and trained to operate the unit, assigned equipment to perform a mission and the assigned equipment being operational.

When a particular system is developed, trade-offs are analyzed to determine the structure of units, the number of units, and the amount of systems needed against a particular threat. Once these are determined, the operational availability of the equipment necessary is determined. It is normally defined by the mean-time-between-failures (MTBF) and the mean-time-to-repair (MTTR) the equipment. As the MTBF decreases or MTTR increases, the operational availability of the system goes down. In order to meet a particular threat, if the unit readiness goes down then the number of units needed goes up. Thus, the threat drives the number of units and their readiness posture.

For any given level of units, readiness and equipment operational availability then, the mean time to repair depends on a spare being available, knowledgeable repairmen, and the design of the equipment for ease of accomplishing the repair (maintainability).

For each system, a Logistics Support Analysis is performed to determine the spares needed at unit level, intermediate levels and depot levels for any given operational availability. As more like end-items are introduced, spares and war reserve requirement for replacements, do not increase linearly. In fact, it can be shown that after a certain number of like computers are located in a unit, additional spares no longer increase as more of the same type are introduced. If different types of computers are introduced, then a complete set of spares is needed for each type at all levels. Likewise, multiple use of the same computer provides the capability for interchange of complete computers to keep the most important systems working in combat to provide continuity of operations.

With proliferation of computer types on the battlefield as currently experienced, operational availability is lower than it would be if there was common hardware with the ISA and interfaces maintained upwardly-compatible.

1.4 Approach

Congressional tasking was approached in a relatively straightforward manner. This is reflected in the organization of the study paper. This Section has described the problem as it impacts the forces, directly and indirectly, on the battlefield.

Sections 2 and 3 delineate the two major domains impacting the embedded computer arena--the military environments and computer technology. These two areas are outside the direct control of the embedded computer decision-makers and act as constraints on embedded computer policy.

In Section 4, near-term and long-term options open to DoD are identified and analyzed. Since many of the alternatives are essentially untried, it is necessary to introduce risk analysis in an attempt to synthesize the cost and benefits of those alternatives for which there is no empirical data. In this manner, a framework is established in which some rational decision-making could be accomplished.

In Section 5 the near-term results of the risk analysis are presented. Both DoD and Service positions are presented.

Section 6 is a blueprint for the future of ISA standardization in DoD. Unlike a standard blueprint, it is three-dimensional, with all three dimensions in a state of change. It can be used as a master plan if the dimensional domains are kept updated, and if decision-makers are ever-conscious of the risk induced by the constant change.

SECTION 2 THE DEPARTMENT OF DEFENSE (DOD) ENVIRONMENT

2.1 The DoD vs Commercial Environment

The total environment of computer usage within the DoD includes a subset common to industry at large and a subset peculiar to the operational environment of the services in their mission of National Defense. This is true of other technologies and functions as well. For instance, some mission requirements of the Air Force Military Airlift Command (MAC) are similar to those of commercial carriers. However, the systems to support the mission of the Air Force include a much broader spectrum of Inter-Continental Ballistic Missiles, Manned Bombers, and Fighters for which there is no direct commercial analog. Similar analogies apply to the missions of the Army, Navy, and Marine Corps.

Just as results of DoD-sponsored research in jet engine technology to obtain lighter weight and higher power for fighter aircraft is applied to commercial aircraft production, so too is defense research in computer and communications technology for strategic and tactical systems applied to the commercial computer industry. The motivation of commercial aircraft manufacturers is sales to commercial carriers, and the motivation of commercial computer manufacturers is private sector sales. The motivation of the DoD is adequate defense against the threat in both cases. Although cost is central, no one advocates sending a commercial aircraft against hostile fighters just because it's cheaper--there is no match. To the extent there is mutual gain, there is reason for DoD to use commercial aircraft, and naturally, during national emergency the commercial airlines will be used where possible. The analogy applies to computers and communications as well.

In the past, the largest part of computer applications within DoD has been for data processing in management information systems for personnel management, financial management, office automation, records management, etc., similar to industry at large. Commercial computer systems, hardware, support software, operating systems, data base management systems, etc., — are used directly as a base for DoD and service application software. In some cases, applications software can be the same. For these systems, DoD is a user of "commercial" computer products just as is industry. Physical space is not normally a problem, the computers need not be mobile, the physical environment can be mechanically controlled, and normal commercial power is available. Though system failure may be costly, interruptions are not a matter of life or death or national security. Spares are not stocked by the government, nor are government personnel generally trained specifically in maintenance of the equipment. Requirements for systems of this type are expected to continue to increase, and DoD will continue to rely on commercial computer products and vendor support.

For computers used in the strategic and tactical units of the Army, Navy, Air Force, and Marines, the DoD is more than just a user: DoD is the owner, operator, and maintainer. Commercial sources of repair are generally not acceptable or available aboard ship, at the battlefield, remote air bases, or accompanying rapid deployment forces. It is in this environment that commercial computers often do not adequately fit the specifications or system requirements. The focus of the remainder of this Section is on this environment.

2.2 Service Operational Missions

The defense missions of the Services are distinct yet overlap in some areas. The Air Force has primary responsibility for strategic aerospace operations to include abilities to deny control of the air to the enemy forces and provide ground and naval forces the assistance necessary for them to control their environment. The Navy has primary responsibility for protection of the sea lanes and in so doing carries aircraft aboard carriers, projects the Marine Corps for amphibious operations, and operates the submarine force. The Army has primary responsibility on land, but is reliant on the Air Force for air transportation and airborne operations and the Navy for sea transport. Each service contributes to the whole.

2.3 Mission Areas

In the functional or mission areas, again there are distinctions and overlaps in functions to be performed. The implementation of functional systems to carry out those missions overlaps because of the different tactical and strategic situations. Broadly, the Mission Areas are:

Army:	Command and control, air defense, fire support, intelligence and electronic warfare, logistics, communications, close combat (heavy), close combat (light), etc.
Navy:	Maintenance of maritime superiority, protection of sea lines of communication, projection of power ashore, providing nuclear deterrence.
Air Force:	Strategic aerospace offense, space operations, strategic aerospace defense, close air support, air interdiction, counter air operations, surveillance and reconnaissance, and special operations.

2.4 The Strategic and Tactical Environment

The strategic and tactical environment will be described in two parts--the operational environment and the support environment--although they too overlap. Much of the support environment is where the equipment is installed and used--the operational environment.

The purpose of a combat system under control of its operators is to detect, track, identify, evaluate and destroy hostile targets within its battlefield sphere. To perform these functions, modern combat systems have, as an integral part, many processing elements. These computers must remain operative throughout a mission. Failure of one of these computers may render a critical sensor or a weapon system inoperative. Equipment malfunctions at a critical moment may cause failure of a mission and loss of many lives. In contrast, failure of a computer in most commercial applications or military business and administrative applications causes temporary inconvenience or, at worst, financial loss.

2.4.1 The Operational Environment

The severe environment in which computers must operate may best be illustrated by one example and then generalized to others. This example, although taken from the Navy, might just as well be an Air Force tactical fighter delivering missiles or warheads to an enemy target, or an Army tactical command post being shelled by enemy artillery or mortars.

2.4.1.1 A Severe Example

The severe environment in which Navy aircraft must operate includes some factors readily apparent and some not so obvious.

Anyone who has seen a launch and recovery of Navy airplanes does not need to be convinced of the severe physical strain put on all systems by a catapult launch and arrested landing. The shock of the steam catapult is diminished to the extent possible, but to accelerate a 30 ton airplane from a dead stop to nearly 150 miles per hour in less than 300 feet requires acceleration rated many times that experienced by commercial aircraft. In addition to the brute acceleration and shock produced by the catapult, the avionics aboard a Navy aircraft must also withstand the tremendous vibration of the jet engines turning at full military power for several seconds just prior to launch. These vibrations are unequaled in the commercial world.

The end of the mission is even more grueling than the beginning. The arrestment of a Navy airplane on a carrier deck must be seen to be appreciated (or believed). The operation has on occasion been described as a "controlled crash". The aircraft literally hooks a large wire which is then payed out at a rate designed to bring the aircraft to a stop in 350 feet! The steel deck is frequently pitching and rolling and consequently the angle at which the aircraft approaches is substantially higher than that used in any other arena. The resulting shock when the aircraft hits the deck is extremely severe. Once the aircraft is "trapped", the throttles are fully advanced to provide the necessary thrust to get airborne in case of a "bolter" or missed trap. This full throttle state introduces another period of intense vibration for the few seconds it takes for the pilot to retard throttles.

The factors described above are readily evident to an observer. In addition to these factors, there are several other environmental conditions which create their own separate destructiveness.

Automobile drivers in the snow belt are aware of the corrosion problem related to salt. Of all environmental requirements leveled on Navy equipment, the exposure to salt is probably the most severe. The tests during Navy qualifications subject the equipment to the same salt exposure it would see in several years of sea service. There is no similar requirement on commercial equipment. Also, the heat and humidity found on a carrier deck in the torrid zone are nearly unbelievable and create conditions rarely seen by commercial equipment. A carrier's planes must always be ready. The ship must be able to launch its air force without extensive preflight. Those same aircraft that have soaked all day in the tropic sun may be airborne in minutes and meeting an enemy at 50,000 feet. The thermal shock alone would be sufficient to damage most commercial equipment. Add to that freezing moisture and it adds up to potential disaster.

In addition, there are "invisible" environments which can cause as many problems as the visible ones. As the aircraft sits on the deck awaiting launch, and during launches and recovery, it is constantly "painted" by the invisible but extremely intense radar beams. Some of these beams contain millions of watts of power and actually cause physical damage to avionics not specifically designed to deal with it.

These factors have only addressed the difficult facts of life related to operations based on aircraft carriers. Things get worse when one considers that the mission of most Navy airplanes is combat. The shock induced by near misses from flak batteries, the intense radiation from enemy jammers, the vibration and shock of gunfire, the acceleration of afterburners, the hi-G turns of air maneuvers, and the absolute necessity for reliable operation combine to levy extreme requirements on the equipment carried aboard Navy airplanes. The job of the Navy aircraft carrier is to project air power anywhere in the world. This will frequently require operation in extremely remote areas where the air groups of the carrier are the only "friendlies" aloft. Navy fighter squadrons seldom practice "one-on-one" or "few-on-few" tactics. They normally assume "one-on-few" or "few-on-many" conditions and must be able to protect the carrier in the face of very stiff odds. Air superiority is achieved to a great extent by superior electronics. Today's carrier-borne fighter, the F-14 Tomcat, carries nearly as much electronics as a Korean War battleship! This complex avionics suite is possible due to weight and size reduction efforts unique to the military arena. Air superiority also requires high reliability and ultimate performance. These qualities are achieved only with the most careful design and Navy requirements are intended to guarantee them to the extent practical.

One does not wish to imply that commercial hardware never achieves high reliability or high performance. One usually finds that a great deal of this high-reliability technology has its roots in high-reliability military hardware built by the same manufacturer. The circuit designs, qualification testing, etc., are all paid for by the Government during the development phase, with the company selling the same or similar hardware on the commercial market at a cost reduced by an amount equal to the qualification tests. The degree of use in the commercial market tends to dictate the commercialization of this equipment. Communications equipment, navigation sets and flight instruments, for example, have significant commercial/military commonality. Computer systems, on the other hand, until recently have not found widespread use in commercial aircraft. One of the main shortfalls of commercial computers is lack of physical and electromagnetic ruggedness. Vibration analysis as well as actual use of commercial minicomputers on non-combat mission aircraft have repeatedly shown that the circuit boards found in such computers cannot withstand the vibration of military aircraft use.

The invisible electromagnetic environment of ship's radar also causes commercial computers to malfunction. The Navy has had a difficult time ruggedizing militarized computers to withstand these beams. A commercial computer would be completely disabled in short order. The intensity of these beams can be appreciated when one considers that test personnel must remember to remove electronic wrist watches during flight deck electromagnetic simulation testing lest the inner working of the watches be destroyed. This is doubly significant since test personnel are required to stay on the fringes on the test area where a personnel radiation hazard does not exist.

The last major factor is logistics support. As big as they are, aircraft carriers are very crowded ships. The available space for spare parts is extremely limited. The Navy requires utilization of the highest-reliability parts obtainable in all equipment in order to reduce as much as possible the repair parts inventory required. The Navy has invested a great effort over the past two decades in the incorporation of common support equipment, that is test equipment that can be used to troubleshoot many different types of equipment. Most commercial equipment would require manual trouble-shooting or expensive adaptation. While the Department of Defense is pressing for system commonality, the fact remains that most commercial computers cannot "converse" without complex adaptation. By specifying common hardware and software, the extensiveness of shipboard support is reduced. Aircraft carriers must be self-sufficient. Remote operations such as the Indian Ocean make support of failed hardware expensive.

These factors combine to place very stringent requirements on Navy electronic systems. The Navy has always sought to minimize the expense of systems by using commercial equipment where possible. The use of commercial navigation radars on yard tug boats and the serious consideration given to commercial weather radar for the reconnaissance version of the P-3 patrol aircraft are good examples. At the present time, however, most commercially-available avionics equipment is not rugged enough to withstand the demanding Naval aviation environment.

2.4.1.2 Less Severe Examples

The SNAP (Ship Non-Tactical ADP) System is an example of commercial computer equipment ruggedized for shipboard use to meet less critical operating requirements than those required of tactical military computer systems. This is possible because SNAP is an off-line ADP data management system, and the life or death of the ship in a battle environment is not directly dependent on its continued operation. The Army's Decentralized Automated Service Support System (DAS3) is another example of a ruggedized computer system used on the battlefield to meet less critical operating requirements. These systems, though business oriented, cannot use "commercial" hardware directly because the equipment will not perform reliably in the shipboard and mobile ground environments.

2.4.1.3 Comparison of Environments

Each of the Services conducts tests of equipment in various conditions and has built an experience base by conducting "post mortems" on causes of microelectronic failure to the component level. For instance, since 1958, NAVSEA has been conducting full scale shock test on every class of combatant. Calibrated underwater tests are employed to simulate combat stress and identify vulnerable items. Table 2-1 lists the relative factors for weighting the application versus environment (MIL-HDBK-217D, 15 Jan 82) while conducting reliability analyses for predicting parts failure. Given any fixed

TABLE 2-1: Application Environment Factor and Description

Environment	Stress Factor	Description
Ground, Benign	0.38	Nonmobile, laboratory environment readily accessible to maintenance; includes laboratory instruments and test equipment, medical electronic equipment, business and scientific computer complexes.
Ground Fixed	2.5	Conditions less than ideal such as installation in permanent racks with adequate cooling air possible installation in unheated buildings; includes permanent installation of air traffic control, radar and communications facilities, and missile silo ground support equipment.
Ground, Mobile	4.2	Equipment installed on wheeled or tracked vehicles; includes tactical missile ground support equipment, mobile communication equipment, tactical fire direction systems.
Space, Flight	0.9	Earth orbital. Approached benign ground conditions. Vehicle neither under-powered flight nor in atmospheric re-entry; includes satellites and shuttles.
Manpack	3.8	Portable electronic equipment being manually transported while in operation; includes portable field communications equipment and laser designations and rangefinders.
Naval, Sheltered	4.0	Sheltered or below deck conditions, protected from weather; includes surface ships communication, computer, and sonar equipment.
Naval, Unsheltered	5.7	Nonprotected surface shipborne equipment exposed to weather conditions; includes most mounted equipment and missile/projectile fire control equipment.
Naval, Undersea, Unsheltered	6.3	Equipment immersed in salt water; includes sonar sensors and special purpose anti-submarine warfare equipment.
Naval, Submarine	4.0	Equipment installed in submarines; includes navigation and launch control systems.
Naval, Hydrofoil	5.9	Equipment installed in a hydrofoil vessel.

TABLE 2-1: (Continued)

Environment	Stress Factor	Description
Airborne, Inhabited, Transport	3.5	Typical conditions in transport or bomber compartments occupied by aircrew without environmental extremes of pressure, temperature, shock and vibration, and installed on long mission aircraft such as transports or bombers.
Airborne, Inhabited, Fighter	7.0	Same as above but installed on high performance aircraft such as fighters and intercepters.
Airborne, Uninhabited, Transport	4.0	Bomb bay, equipment bay, tail, or where extreme pressure, vibration, and temperature cycling may be aggrevated by contamination from oil, hydraulic fluid and engine exhaust. Installed on long mission aircraft such as transports and bombers.
Airborne, Uninhabited, Fighter	8.0	Same as above but installed on high performance aircraft such as fighters and intercepters.
Airborne, Rotary Wing	8.5	Equipment installed on helicopters; includes laser designators and fire control systems.
Missile, Launch	13.0	Severe conditions related to missile launch (air and ground), and space vehicle boost into orbit, vehicle re-entry and landing by parachute. Conditions may also apply to rocket propulsion powered flight.
Cannon, Launch	220.0	Extremely severe conditions related to cannon launching of 155mm and 5-inch guided projectiles. Conditions apply from launch to target impact.
Undersea, Launch	11.0	Conditions related to undersea torpedo mission and missile launch.
Missile, Free Flight	3.9	Missiles in non-powered free flight.
Airbreathing Missile,	5.4	Conditions related to powered flight to air breathing missile; includes cruise missiles.

quality level of part, the environment factor is a multiplicative factor for expected failures vs. application and is used to calculate parts-reliability or expected-failure-rate. The larger the factor, the more failures are expected.

2.4.2 Requirements of the Operational Environment

The Services constantly review the threat against which they will operate. Studies are performed to balance the structure of units, the number of units, the systems and their number, and the system requirements to counter the threat. New system developments or modifications of old systems are based on these studies. Total life-cycle costs, not just initial acquisition costs, are considered. In fact, the majority of costs come in the operation and maintenance phase, particularly for software-intensive systems.

The reliability of equipment and its operational availability directly effect the number of units needed. Once the number of units, their doctrine and tactics, and the number of systems are determined to meet the threat, the readiness goal of the unit is set. Naturally, the readiness of the Services to perform their mission is negatively impacted by the down-time of their systems and equipment.

The engineering requirements derived from the operational scenarios are placed in a system specification and an acquisition stiategy is developed. Some of the specifications may be developed by testing during the system development. A discussion of some of the requirements follows:

- Enclosures
- Survivability and Vulnerability
- Interoperability
- Interchangeability
- Reliability
- Maintainability
- Hardening
- Vibration, Acceleration, Shock
- Thermal
- Power, Size, and Weight

2.4.2.1 Enclosures

Enclosures include man-packs, projectiles, torpedoes, missiles, tracked and amphibious vehicles, mobile and air-droppable shelters, aircraft, ships, submarines, buildings, trailers, etc. Exposure and protection afforded the equipment derives from where and how it is to be employed (see Section 2.4.1.3).

2.4.2.2 Survivability and Vulnerability

Defense systems must be designed to operate in hostile environments--both man-made (e.g., laser or nuclear weapons, etc.) and natural (e.g., dust, rain, or natural space radiation, etc.). The survivability of a system is its capability to avoid or withstand these hostile environments without suffering degradation or malfunction which could impair its ability to accomplish its designated mission. Specifically, survivability includes the ability to withstand the effects of blast, heat, radiation, electromagnetic pulse (EMP), friendly and enemy electromagnetic interference, etc., and the ability of the system or unit to fail or repair itself in a predicted manner and to be reconstituted. Computer systems in strategic and tactical units, particularly in networks, and their software are specially designed to overcome these known effects. Vulnerability is a measure of the system's lack of resistance to these known effects. Unit survivability is directly related to the functioning of its key systems. Interchangeability of computers and parts assists in survivability of key systems.

2.4.2.3 Interoperability

In a broad sense, interoperability can be confused with hardware interchangeability. Generally, interoperability, when used in connection with systems which exchange data refers to the ability of two systems to exchange data and understand the relationships between these data objects. An approach used in DoD achieves interoperability by exchanging data in "messages" of pre-defined formats which allow data elements to be recognized and processed automatically. The future requires that systems in the strategic and tactical environment operate as a network of systems. Commanders must have accurate and timely data on which to base decisions and fight their units. Common hardware and software greatly facilitates interoperability. Open and closed system network architectures, will greatly facilitate interoperability and reduce the amount of different software to be fielded in command and control and sensor systems that must exchange data. However, data in common message formats can be exchanged over data links by systems with dissimilar hardware and software.

2.4.2.4 Interchangeability

For some time, the Services have acquired common items that allow some degree of interchangeability. This is true with tires, engines, trailers, etc. They are procured competitively to specifications which allow interchangeability--the ability to substitute one item for another. Common ammunition of dissimilar weapons is an example.

Because commercial computer manufacturer's instruction set architectures are proprietary and protected, there has been relatively little interchangeability of computers in defense systems except where identical computers are used (generally from the same production line and within the same system). This has also limited competition (the ability to compete) for like items. For systems at sea and on the battlefield, being able to interchange LRUs (line-replaceable units) will enhance the survivability of units by allowing the most important systems to be kept in operation even when resupply cannot be accomplished.

2.4.2.5 Reliability

Perhaps the most important consideration in hardware design is its reliability. Reliability is the probability that a part, assembly, equipment (i.e., box), or system will perform for a specified interval under stated conditions with no malfunction or degradations that required corrective maintenance action. Reliability considerations are addressed at the earliest stages to establish design objectives. This process begins with the selection of piece-parts that will be used in hardware.

Through continued analysis of failures, a complete set of tests and specifications has been developed over the years to assure that electronic parts meet the requirements of the environment.

Components procured to these specifications are referred to as MIL-SPEC parts. Table 2-2 is reproduced from MIL-HDBK-271D (Jan 82) to point out two significant factors: One, there is a complete set of specifications from which to choose, so that over-qualified parts are not specified. Two, the effective reliability is greatly increased with additional screening, temperature, and vibration testing and burn-in to remove infant mortality. Note that the higher the quality factor, the more likely a part will fail.

TABLE 2-2: Quality Factors

Quality Level	Quality Factor	Description
S	0.5 (Best)	Procured in full accordance with MIL-M-38510, Class S requirements.
B	1.0	Procured in full accordance with MIL-M-38510, Class B requirements.
B-0	2.0	Procured in full accordance with MIL-M-38510, Class B requirements, except that device is not listed on Qualified Products List (QPL). The device shall be tested to all the electrical requirements (parameters, conditions and limits) of the applicable MIL-M-38510 slash sheet. No waivers are allowed except current and valid generic data * may be substituted for Groups C and D.
B-1	3.0	Procured to all the screening requirements of MIL-STD-883, Method 5004, Class B and in accordance with electrical requirements of MIL-M-38510, DESC (Defense Electronic Supply Center) drawings, or vendor/contractor electrical parameters. The device shall be tested to all the quality conformance requirements of MIL-STD-883, Method 5005, Class B. No waivers are allowed except current and valid generic data* may be substituted for Groups C and D. This category applies to DESC drawings and contractor prepared specification control drawings (SCDs) containing the above B-1 screening and quality conformance requirements.
B-2	6.5	Procured to vendor's equivalent of the screening requirements of MIL-STD-883, Method 5004, Class B, and in accordance with the vendor's electrical parameters and vendor's equivalent quality conformance requirements of MIL-STD-883 Method 5005, Class B. Applies to contractor prepared SCDs containing the above B-2 screening and quality conformance requirements.

TABLE 2-2: (Continued)

QUALITY LEVEL	QUALITY FACTOR	DESCRIPTION
C	8.0	Procured in full accordance with MIL-M-38510, Class C requirements.
C-1	13.0	Procured to screening requirments of MIL-STD-883, Method 5004, Class C and the qualification requirements of Method 5005, Class C. Generic data may be substituted for Groups C and D.
D	17.5	Hermetically sealed part with no screening beyond the manufacturer's regular quality assurance practices; parts encapsulated with organic material.**
D-1	35.0 (Worst)	Commercial (or non-mil standard) part, encapsulated or sealed with organic materials (e.g., epoxy, silicone or phenolic).

* Group C generic data must be on data codes no more than one year old and on a die in the same microcircuit group (see appendix E of MIL-M-38510) with the same material, design and process, and from the same plant as the die represented. Group D generic data must be on data codes no more than one year old and on the same package type (see Section 2.1.3.12 of MIL-M-38510) and from the same plant as the package represented.

** All encapsulated devices must be subjected to 160 hr. burn-in at 125°C, 10 temperature cycles (-55°C to 125°C) with end point-electricals, and high-temperature continuity test at 100°C.

The additional testing for MIL-SPEC components is not without cost; however, this cost is returned during the life cycle in purchase of less spares and having equipment that meets the required operational availability of equipment. Normal commercial testing is not sufficient to demonstrate the reliability required in the operational environment.

Commercial chips are initially designed, for the most part, for commercial functions and environments. If the device is a new technology, new manufacturing methods and equipment are used and there is a period of initial "shake-down" use. Some semiconductor technologies are better suited to military operational environments. If the device is to withstand the more rugged environments, it may require redesign, but the basic processes from "CAD-CAM" (Computer Aided Design-Computer Aided Manufacturing) design--wafer fabrication, probing, scribing, wafer-breaking, cleaning, die mounting, wire mounting, wire bonding and wire pull are basically the same for commercial and military systems. At this point, for rugged requirements, the type carrier is non-porous, the process inspections increase, the chips are cycled over the temperature range with end-point electrical tests and the chips are burned in to reduce infant mortality. Experience has shown failures to decrease greatly with this process (see Table 2-3).

TABLE 2-3: Cost/Degrees of Testing

Type Qual		Failure Rate Multiplier	100 UP Cost
Commercial		35.0	$10.00
Commercial	(with burn-in)	17.5	
MIL-SPEC	(Class C - no burn-in)	8.0	$19.00
MIL-SPEC	(Class B - with burn-in Vendor Criteria)	6.5	$22.00
MIL-SPEC	(Class B - with burn-in JAN Criteria)	1.0	$61.40

In spite of these additional costs, the life-cycle cost of computers goes down and the operational availability goes up with fewer failures. The other engineering design costs for temperature, humidity, TEMPEST, nuclear survivability, vibration, LSA and data, plus testing of the computer for its intended environment overshadow these costs. These costs are repaid in increased operational availability and reduced sparing costs, as shown later.

The environmental factors directly affect the number of failures of chips. As an example, the equation for predicting the number of failures (λ_P) failures per 10^6 hours is (for a monolithic bipolar and MOS random logic LSI and microprocessor devices equal to or greater than 100 gates):

$$\lambda_P = \Pi_Q [C_1 \Pi_T \Pi_V \Pi_{PT} + (C_2 + C_3)\Pi_E]\Pi_L \text{ WHERE}$$

Π_Q = QUALITY FACTOR (SEE TABLE 2-2)

Π_T = JUNCTION TEMPERATURE FACTOR

Π_V = VOLTAGE FACTOR

Π_{PT} = ROM AND PROM PROGRAMMING TECHNIQUE FACTOR

Π_E = ENVIRONMENT FACTOR (SEE TABLE 2-1)

Π_L = LEARNING CURVE FACTOR

C_1, C_2, C_3 ARE EMPIRICAL COEFFICIENTS BASED ON THE COMPLEXITY OF THE DEVICE.

The following specific example is given on page 5.1.2.6-1 of MIL-HDBK-217D:

Description: An 8192 N-channel MOS UV-EPROM in a Ground Fixed application, junction temperature of 55°C, procured to vendor equivalent B-2 Quality level. The production line has been in continuous production. The device is a ceramic/metal DIP (Dual-Inline Proccessor), solder hermetic package with 24 pins.

Factors: The following factors are taken from various tables of the above-mentioned handbook. The tables for environmental and quality factors are shown in this document (Tables 2-1 and 2-2, respectively).

Π_Q **(QUALITY FACTOR B-2) = 6.5**

Π_E **(BENIGN ENVIRONMENT) = 0.38**

Π_E **(GROUND FIXED ENVIRONMENT) = 2.5**

Π_E **(GROUND MOBILE ENVIRONMENT) = 4.2**

Π_T **(JUNCTION TEMPERATURE FACTOR) = 0.71**

Π_V **(VOLTAGE FACTOR) = 1.0**

Π_L **(LEARNING CURVE FACTOR) = 1.0**

Π_{PT} **(PROGRAMMING TECHNIQUE FACTOR) = 1.56**

$C_1 = 0.055$

$C_2 = 0.0024$

$C_3 = 0.009$

λ_p(Ground Fixed) = 6.5 [(0.55)(0.71)(1.0)(1.56) + (0.0024 + 0.009)(2.5)] 1.0
= .58 failures in 10^6 hours.

λ_p(Benign) = 6.5[0.055)(0.71)(1.0)(1.56) + (0.0024 + 0.009)(0.38)]1.0
= .424 failures in 10^6 hours.

λ_p(Ground Mobile) = 6.5[(0.055)(0.71)(1.0)(1.56) + (0.0024 + 0.009)(4.2)] 1.0
= .7072 failures in 10^6 hours.

Therefore, the percentage increase from the benign to the ground mobile with all other factors held constant is approximately 67%. The percentage increase from ground fixed to ground mobile is approximately 22%.

It is often stated that commercial computers are becoming more reliable and should be used directly. They are becoming more reliable, but the operational environment degrades their reliability even at the chip level. Merely increasing the level of integration does not improve this reliability linearly. In fact, adding circuit complexity is a tradeoff with reliability. Some improvement in reliability is obtained by using computers with fewer chips, however, experience to date shows that about the same number of chips are used because of increased requirements in most applications.

2.4.2.6 Maintainability

Maintainability is a characteristic of design and installation which is expressed as the probability that an item will be retained in or restored to a specified condition, within a given period of time, when the maintenance is performed in accordance with prescribed procedures and resources. Engineering for maintainability must also consider how often maintenance tasks must be performed, the skill level of the maintainer, and the time required to do the job. It has a direct bearing on the total system's life-cycle cost. Acquisition costs are one-time costs; maintenance costs are recurring.

The maintainability parameter is a measure of ease with which a system can be repaired or maintained. Commercial technology focuses upon equipment which is repaired at the factory or on the customers' premises. The Services must train maintainers and plan for maintenance under operational conditions. Recall that for most of these systems, the Services must have the capability to restore its systems to operation. They are the owner, operator, and maintainer. Military systems are repaired as far forward in the battle area as possible. If the equipment cannot be repaired, it must be evacuated and replacement brought forward. This requires diagnostics and fault isolation to the line-replacable unit (LRU) at the Unit level. This affects design, reliability, and need for test equipment. This drives a need for built-in-test capability for computers used in deployable systems.

2.4.2.7 Hardening (Radiation, EMI, EMP, TEMPEST)

Hardening is the measure of the ability of a system to withstand exposure to one or more of the effects of either nuclear or non-nuclear weapons and other environmental factors. Thus, the vulnerability of a system is a measure of its hardness when exposed to those threatening environments. The design engineers must take the system's vulnerabilities into account and minimize them by hardening techniques.

Hardening against nuclear effects starts at the piece-part level. Parts are used which are characteristically less susceptible to radiation effects. As an example, junction-isolated integrated circuits are used because they are less susceptible to radiation effects than other integrated circuits. Majority carrier devices, such as field effect transistors, are used in lieu of bipolar transistors where possible because the decrease in minority carrier lifetime caused by radiation does not severely degrade its performance. After piece-part selection, the next step in designing hardened equipment is proper circuit design. The circuits should be designed to take into account the unavoidable but predictable degradation caused within the piece-parts by radiation exposure.

Hardening for non-nuclear effects can be provided in a number of ways. Blast effects are reduced by providing redundant structures and designing the system with a high degree of stress and structural safety margins. Thermal and laser effects can be compensated for by proper use of materials, processers, skin thickness, and thermal coatings (e.g., reflective paint, etc.).

The threat from visual, infrared, audible and electromagnetic detection systems can also be minimized. Camouflage or contrast-reducing paint, low emission metals and coatings and noise reduction techniques help. Electromagnetic detection can be overcome by reducing radar cross-sections, using radio frequency absorbing materials and paints, and providing electronic countermeasure (ECM) equipment to jam detection systems. Based on the system's mission, a combination of methods must be used to address the variety of threats to establish appropriate level of hardening.

2.4.2.8 Vibration, Acceleration, Shock

The physical location and use of computers in tactical and strategic systems determines the specifications for vibration, acceleration and shock that must be endured. There are wide variations within the same system and across systems derived from the intended use and the systems environmental requirements. For instance, a missile system may include the missile itself with extensive computers for control and guidance, a launch vehicle, ground control stations and command and control computers.

Battlefield computers are expected to operate in helicopters, missiles, self-propelled guns, tanks, armored personnel carriers used as tactical command posts, open vehicles for resupply or troop movement, and to withstand blasts from mines from opposing artillery which otherwise does not render the carrier inoperative. Army equipment is expected to pass vibration tests representative of that environment such as:

(1) 40 Gs for 11 milliseconds,
(2) impact tests in transit cases,
(3) 2 Gs for over 5 to 2,000 hertz.

Aboard naval ships, computers are subjected to repeated shocks of a ship moving through high seas, severe shock resulting from naval gunfire, underwater explosions (mines and torpedoes) and continuous structural vibration from the ship's propellers and proximity to other equipment.

In aircraft and ICBMs (Intercontinental Ballistic Missiles), there are a different vibration spectra, acceleration and shock environments. Aircraft operated aboard carriers and assisted by catapult derive a different set of specifications than those whose function is transport. Jet engines with afterburners add to the severity of requirements.

2.4.2.9 Thermal Requirements

Thermal requirements are derived both from heat generated by the equipment itself and the ambient in which the equipment must operate. The temperature requirement is one of the most severe and related directly to reliability and future performance of the equipment. These requirements are obtained from the mission profile of the system. Thermal requirements are compounded by other requirements. For instance, the requirements of dust and humidity, both of which require the unit to be sealed to prevent shorts, prevent the majority of strategic and tactical systems from using equipment which is cooled by blowing ambient air over the components. Likewise EMI, EMP and TEMPEST requirements require shielding. Sometimes liquid nitrogen, water or conditioned air is necessary, but then the functioning of auxiliary cooling systems affects the reliability and operational availability of the equipment.

Operational computers must be mechanically designed to meet thermal shock resulting from rapid changes of temperature within its normal operating range. Coefficients of expansion of base materials must be similar to prevent the minute connections of integrated circuits from breakage and failure.

2.4.2.10 Size, Weight, and Power

The environment of strategic and tactical systems places exacting demands on these critical factors. In aircraft, size, weight, power and form factors exact a price from other components of the system. Higher power requires more generators, weight more thrust, and size becomes important in the cockpit. For the Army which must carry generators, operate from vehicular-supplied power (and provide batteries for hand-held devices and generator failure) or other devices, size, weight, and power exact a toll in requirements for other items. Commercial power, though expensive, it not a critical factor in the design of commercial computers. With the advent of VLSI, these factors are continuing to diminish in importance, but will always be more important for tactical systems. Tactical computers must be designed to operate in a predicted fashion with the inevitable power interruptions which occur. Failures in real-time systems cannot be allowed to destroy programs in memory or disrupt data bases where power failures occur nor compute erroneous results as power returns.

2.4.2.11 Summary

The operational environment of strategic and tactical units has many added requirements. Commercial computer hardware cannot be used directly as is because of increased failures caused by the environment. DoD, through failure analysis and testing, has quantified the effects of the environment on various components. Varying degrees of militarization are required based on the systems application.

2.5 Maintenance Concepts vs Operational Requirements

The development of any DoD system is guided principally by system operational requirements, the development of which is derived from the threat analysis as pointed out previously. The Statement of Operational Need (SON) establishes the minimum essential operational, technical, personnel, training, logistic and requirements needed to develop, support and operate the system. Therefore, the requirements for the development and/or acquisition of all computer resources for deployable units, whether as components of weapon systems, communications and command and control systems, or as separate end items, originate as operational requirements. The maintenance and support requirements are then determined by operational requirements and all system maintenance and support must conform to the concepts and organizations developed for the Service units which will employ the system.

2.5.1 The Logistics Environment

From the operational environment discussion it was concluded that there is a broad spectrum of computer usage in deployable units. Size, location, operational environment, support environment, and application all affect equipment and maintenance process. Certain classes of systems seem to fit a particular maintenance concept but there are always exceptions even within the same system. For instance, on the battlefield, a computer specified for a missile system may provide information to a ground station in a van. The results of a failure and the environment are different for each. In the avionics environment there are broad statements which can be made about Air Force aircraft being deployed generally in one type unit and centralized maintenance being the preferred concept. Yet, there are many different kinds of aircraft: spacecraft, transports, fighters, helicopters. Take helicopters: between the Services helicopters are used for completely different purposes. The Army has attack helicopters, used in close ground support (and others), the Navy uses helicopters at sea, and the Air Force may have to travel great distances to retrieve downed pilots. Where possible, the Services combine their requirements for economy. For space and weight requirements, avionics systems have tended not to standardize form factor. For economics where possible, support software and instruction set architectures have been specified. Aircraft, in general, do return to a central point for maintenance.

Aboard ship where space is limited, form factors and spares have been specified by the Navy because of the Logistics environment. This is amplified in Section 4. A ship's mission length is extremely long compared to that of a fighter.

The Army currently has a proliferation of computer types for the battlefield and is moving to reduce that proliferation for economies of scale. The Army carries its support with it in general and repairs equipment on site when feasible due to the fact that supply and transportation lines are often severed during battle conditions. Without performing repairs on-site, operational availability could not be maintained. The Divisions of the Army have most types of systems in the Army inventory, and proliferation increases the size of the support units required. The benefits from reducing this proliferation are amplified in Section 4. It should be noted that Army wartime systems are maintained by "green suit" personnel while many of the peacetime systems use civilian contractors.

The operational environment in which a specific computer system is to operate significantly impacts the maintenance concepts which will be employed to support the system. Those weapon systems used in forward areas, such as components of tanks, aircraft, ships, artillery or battle management systems, may require maintenance by "black box" (LRU) replacement. This provides effective organizational maintenance and allows the components to maintain environmental and nuclear hardening integrity until they can be removed to a maintenance center with the required equipment and environment to "open" these components. The size of the LRU must be traded off against the cost of spares. Conversely, some computers cannot be moved to a maintenance facility and require diagnostic equipment, and spares on location to allow fault isolation and component replacement down to the printed circuit and/or "plug-in" module. Most tactical computers make extensive use of BIT/BITE which locates and isolates faults down to a group of boards or to a single board level.

2.5.2 Logistic Considerations

The LSAR is a data system for documenting the Logistic Support Analysis (LSA). It is comprised of analysis worksheets, programs, computer-generated outputs, and associated instructions. It provides input to technical publications, manpower requirements, training programs, support equipment, and Test, Measurement and Diagnostic Equipment (TMDE) requirements. It also provides for transportation and transportability requirements, facilities requirements, and provisioning technical documentation. The LSA is begun at system initiation and kept current through the life of the system based on overhaul programs, producibility engineering and planning, sample data collection, Modification Work Orders, product improvements and field data feedback.

While the basic logistics concepts of any system are derived from the operational requirements, the specific detailed logistics support data comes from the Logistic Support Analysis (LSA) process. The LSA process, described in MIL-STD-1388-1/2 (and other Service regulations), is the logistic analytical effort implemented within the system engineering process. It includes the use of the necessary analytical tools and models to:

- Develop and evaluate alternative support concepts.
- Project manpower and personnel impact.
- Perform tradeoffs between system design and ILS (Integrated Logistics Support) elements and tradeoffs among ILS elements.
- Integrate support planning and design.
- Measure Life-Cycle Cost (LCC) impact of materiel and support system alternatives.

The LSAR also serves as a checklist of the decisions required to identify support requirements and evaluate the supportability of system/equipment design.

2.5.3 Maintenance Concepts

Table 2-4 and Figure 2-1 show various concepts for keeping equipment in operation. The equipment location may be a ship, airplane, a self-propelled gun, or a command and control shelter capable of deployment on a moment's notice. Regardless of the repair or replace concept, enough spares must be available to keep the equipment in operation for the duration of the mission assigned to the unit (see discussion of Operational Availability, Section 2.5.4.1). Spares are then stocked at various unit locations depending on the most cost effective maintenance concept for the system.

Recall that the Services become the owner, operator, and maintainer of these systems. During the initial phases of system introduction, contractor support may be employed, new equipment teams visit the units and conduct training. Later schools are set up to train soldiers, sailors, and airmen as replacements for those who will rotate from the operational units. Spares are then stocked in supply channels and normal maintenance procedures take over.

TABLE 2-4: Maintenance Concepts

1. Discard LRU at failure.
2. Replace modules at ORG. Scrap bad modules.
3. Replace modules at INT. Scrap bad modules.
4. Replace modules at Depot. Scrap bad modules.
5. Replace modules at ORG. Repair modules at INT.
6. Replace modules ar ORG. Repair modules at Depot.
7. Replace parts at INT.
8. Replace modules at INT. Repair modules at Depot.
9. Replace parts at Depot.
10. Replace parts at ORG.
11. Replace modules at Equipment. Scrap bad modules.
12. Replace modules at Equipment. Repair modules at ORG.
13. Replace modules at Equipment. Repair modules at INT.
14. Replace modules at Equipment. Repair modules at Depot.
15. Replace modules at Contractor. Scrap bad modules.
16. Replace modules at Equipment. Repair modules at Contractor.
17. Replace modules at ORG. Repair modules at Contractor.
18. Replace modules at INT. Repair modules at Contractor.
19. Replace parts at Contractor.
20. Recheck LRU at ORG. Scrap bad LRU.
21. Recheck LRU at ORG. Replace modules at INT. Scrap bad modules.
22. Recheck LRU at ORG. Replace modules at Depot. Scrap bad modules.
23. Recheck LRU at ORG. Replace modules parts at INT.
24. Recheck LRU at ORG. Replace modules at INT. Repair modules at Depot.
25. Recheck LRU at ORG. Replace parts at Depot.
26. Recheck LRU at ORG. Replace modules at Contractor. Scrap bad modules.
27. Recheck LRU at ORG. Replace modules at INT. Repair modules at Contractor.
28. Recheck LRU at ORG. Replace parts at Contractor.
29. Replace LRU at ORG. Repair System.
30. Replace PCB (in LRU) at Intermediate Forward. Repair LRU.
31. Repair PCB at GS, EAC or Depot. Replace components.

FIGURE 2-1: Maintenance Concepts

```
/ Concept  /  Equipment / Organization / Intermediate /  Depot  / Contractor /

     1.          E(D)
     2.          E--------->E-->M(D)
     3.          E------------------------->E-->M(D)
     4.          E-------------------------------------->E-->M(D)
     5.          E--------->E------------->M-->R
     6.          E--------->E-------------------------->M-->R
     7.          E------------------------->E-->R
     8.          E------------------------->E----------->M-->R
     9.          E-------------------------------------->E-->R
    10.          E--------->E-->R
    11.          E-->M(D)
    12.          E--------->M-->R
    13.          E------------------------->M-->R
    14.          E-------------------------------------->M-->R
    15.          E------------------------------------------------->E(D)
    16.          E------------------------------------------------->M-->R
    17.          E--------->E------------------------------------->M-->R
    18.          E------------------------->E---------------------->M-->R
    19.          E------------------------------------------------->E-->R
    20.          E--------->E(D)
    21.          E--------->E------------->E-->M(D)
    22.          E--------->E-------------------------------->E-->M(D)
    23.          E--------->E------------->E-->R
    24.          E--------->E------------->E----------------->M-->R
    25.          E--------->E-------------------------------->E-- R
    26.          E--------->E------------------------------------->E-->M(D)
    27.          E--------->E------------->E---------------------->M-->R
    28.          E--------->E------------------------------------->E-->R

Legend:   E = Equipmemt or LRU          R = Replaced at that location
          M = Module                   (D)= Discard
```

Note that regardless of the maintenance concept (which is developed for cost effectiveness), two things are important:

(1) Replacement spares must be available to keep the system in operation even if the failed part is discarded, and
(2) Reliability of the equipment and mission duration determine the number of spares to be stocked.

Even if contractor repair of the failed equipment or spares is chosen, the readiness of the unit and operational availability of the equipment is determined by the availability of spares and whether in peacetime or at war.

At the equipment site, the time to return a system to an operational state depends on the availability of trained maintenance personnel, the maintainability of the equipment, and the availability of spares. Note that if a part is modified, the operator training, the Built-in-Test capability (diagnostics) and any automatic test equipment along the repair chain must be modified and the system purged of old spares. It is these costs that offset the costs of making changes to stay up with the latest technology, as long as the original system meets the requirement.

2.5.4 Maintenance Concept Drivers

During the development of any maintenance concept, the "drivers" stem from:

- The operational availability required of the system,
- The environment of system maintenance, and
- Cost effectiveness of sparing and repair concept.

2.5.5 Operational Availability

Operational Availability (A_o) is defined as the probability that, when used under stated conditions in an actual operational environment, a system will operate satisfactorily at any time. It is expressed as:

$$A_o = MTBF/(MTBF + MLDT)$$

where MTBF is the mean-time-between-failures (expressed in hours) and MLDT is the mean-logistic-down-time, which includes all logistic down time resulting from maintenance, supply, transportation and administrative actions.

When one reviews the current state-of-the-art for computer devices, it is apparent that the solid state components currently used, when coupled with available manufacturing techniques, and the redundant circuitry can (and does) provide increases in MTBF compared to older electronic devices. Table 2-5 depicts the A_o achievable with MTBFs of 1,000 to 10,000 hours and MLDT of 0 to 90 days.

TABLE 2-5: A_o vs. MLDT

	MTBF (hours)			
MLDT Da.-Hrs.	10k A_o	5K A_o	2.5K A_o	1K A_o
5-120	.988	.976	.954	.892
15-360	.965	.932	.874	.735
30-720	.932	.874	.776	.581
45-1080	.902	.822	.698	.480
60-1440	.874	.776	.634	.409
75-1800	.847	.735	.581	.357
90-2160	.822	.698	.536	.316

While it is apparent that the steadily increasing MTBFs available through current technology can make significant improvement in Ao, it is also noticeable that the MLDT must be kept low. Small decreases in computer Ao can have significant impact on system availability. Also of importance here, as the number of computer models increase, the number of separate repair parts, different maintenance skills, different fault isolation techniques, maintenance manuals and test devices will increase.

The operational availability of the major weapons and support systems are determined by the users' operational requirements. System operational availability is determined by the individual failure rates and downtime of its parts. This, in turn, statistically drives the operational availability and reliability requirements of embedded computer devices even higher.

Reliability is by nature expensive, but this is essentially a "designed-in", "front-end", one-time development expense, which is significantly offset by the maintenance and supply cost avoidances which will be achieved throughout the system life cycle.

Logistic responsiveness (which determines MLDT) in the operational availability equation, is composed of a number of subfactors, all of which are also cost drivers. These include:

- Provisioning -- the process of having the right repair parts, in the right quantities, at the proper location is the most critical, and most expensive of the cost drivers.

- Maintainability -- which must be designed into the system. This includes the development of adequate built-in-test-equipment (BITE) to fault-isolate the system, the use of modular construction techniques to facilitate repair, and the provision of the means to access replaceable components and test points.

- Personnel and training -- paramount to maintaining logistic responsiveness will be having the appropriate numbers of adequately trained personnel available for maintenance and supply operations for each system.

The maintenance concept and sparing costs depend on the potential density of computers as well as the reliability of the computers. There are a number of intangible factors such as "maintainer familiarity"--that is, people who operate the equipment and know how to fix it--training, user-machine interface, etc.

The compelling reasons for embedded computer hardware standardization for certain mission areas in the Army and Navy are operational availability, increased capability, and economics.

As stated before, once the number of units, their structure, and weapons are determined, the operational availability (readiness) of units and complete systems is determined by the operational availability of its equipment.

The standard criterion for most Navy systems is self-sufficiency over a 90-day mission. Perhaps the most complex environment relative to resupply of failed parts is the case of the Ballistic Missile Submarine. Here completion of an assigned mission, critical to the national defense posture, requires that this particular weapons platform sustain itself for continuing periods of up to six months or more. This sustaining capacity must be obtained within constraints of a compacted submerged vessel, restricted severely in both space and the number of personnel carried. Standardizaton of computer resources, with its inherent commonality of on-board components and maintenance procedures, is a force multiplier, thereby producing efficient utilization of scarce space and skill level resources. Additional spares and personnel skills cannot be brought in to augment core capabilities because of the absolute necessity of the SSBN to remain in a covert status. Compromising capabilities of tactical units by relaxing essential requirements is not an alternative.

Mean time to requisition spares turns out be very critical. It can vary from a few minutes (if the spares are readily available) to up to 500 hours or more (if spares have to be flown, shipped, or convoyed in). This actual time can vary extensively around the mean. Consequently, high operational availability is not only a function of reliability but also, more importantly, how fast a failed unit can be restored to operations. This is strongly dependent on the availability of spares or cannibalizing less critical systems to keep the most important ones operating if hardware is common.

Specifications associated with the Navy's standard mainframe computer, the AN/UYK-43, establish that the mean probability of continued operation without failure during a 90-day mission is 70 percent. Obviously, repair provisions must be made to attain a reasonable operational availability. Table 2-6 indicates that when no spares are carried on-board, repair will require around 500 hours and the operational availability will only be 0.879.

TABLE 2-6: Available Spares Affect A_o and Cost

	No On-Board Spares	50% of LRU Types On-Board	One Spare for Each LRU Type On-Board
Operational Availability	0.879	0.968	0.998
Spared Quantity	0	27	54
On-Board Spares Cost	0	$81,250	$162,500

A 50 percent availability of on-board spares will yield 10 percent improvement in mission availability at a cost of $81,250. High-end sparing of one spare for each logistically replaceable unit will bring the computer almost to the 0.999 availability goal. This is only achieved at a cost in spare parts carried on-board approaching one-half the original cost of the computer. The 6000-hour specified reliability of the AN/UYK-43 and -44 yields a 0.30 chance of its failing during a 90-day mission or an Ao of 70%.

Installation of more than one computer of the same type on board the same ship modifies spares and related cost analyses. Derivation of this impact can be made by first establishing the cost of on-board spares for a single computer system. Figure 2-2 shows a sparing curve representative of the AN/UYK-43 and AN/UYK-44. Examination of the low end, with one computer on board, indicates that approximately 50 percent of the line-replaceable unit (LRU) types are authorized to be carried on board. These spares are termed "technical override spares". They are the minimum number of spares to ensure that in the event of failure, computer programs can be loaded and the maintenance processor can be made to run so that the computer has the capability to diagnose itself. This basic set of "technical override spares" must be on board, whether the ship carries only one or fifty of the same computer. Consequently, an additional unique set of "technical override spares" is needed to support each different computer type carried on board.

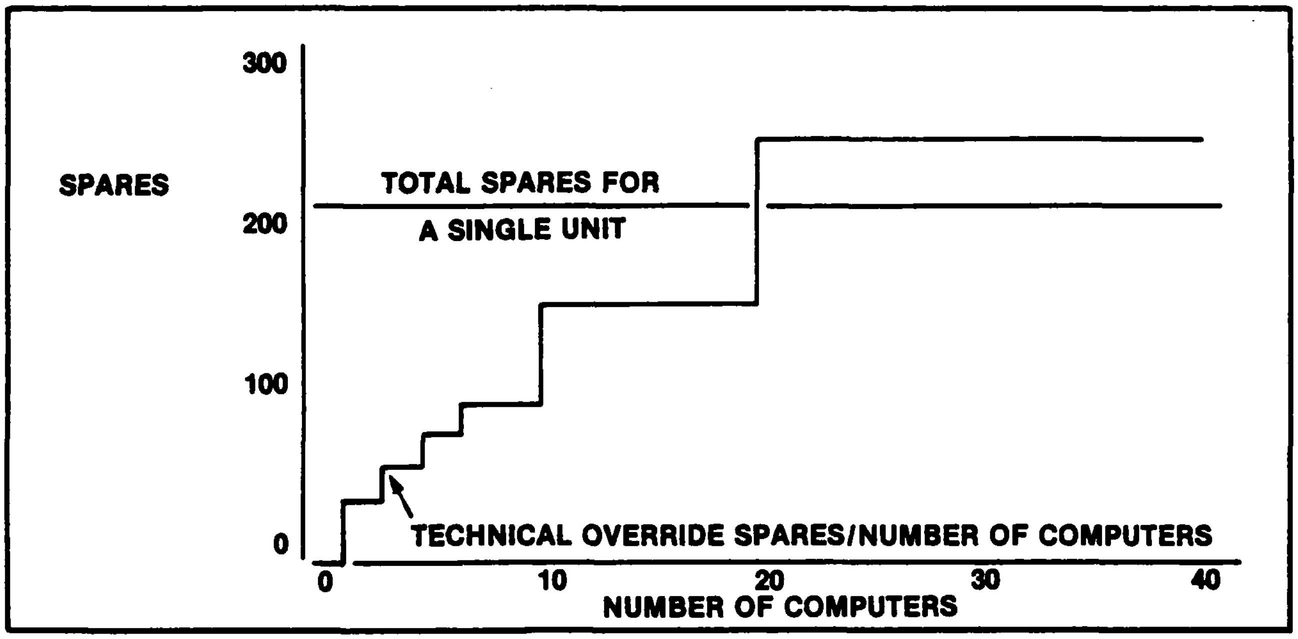
The cost of these minimal spares (shown on the bottom of Figure 2-2) is based on a ten-year projection of the use of the AN/UYK-43 mainframe and the AN/UYK-44 minicomputer. Using the population assumptions of 1500 AN/UYK-43s on 250 ships and 20,000 AN/UYK-44s on 500 ships, the very minimum (assuming only two basic types of computers from the same family) fleet-wide technical override sparing cost will be approximately 30 million dollars. Obviously, as the number of unique types of computer increases, the cost associated with the minimum sparing requirements increases proportionally. It should also be noted that this minimal "technical override sparing" is still not sufficient to satisfy the needed A_0 of 0.999. Computer proliferation would levy an unacceptable economic burden on the Navy based solely on the cost of satisfying fleet-wide sparing requirements. Additional costs associated with program elements such as support software, training, in-service engineering, and the loss of large volume production cost breaks further exacerbate the already unacceptable economic burden.

The Army has studied the proliferation problem for some time. The Military Computer Family (MCF) program was undertaken to derive the benefits from competition with a vendor-independent ISAs and to provide common hardware for as many systems as possible. The common ISA allows competition for hardware at different levels of hardening, while preserving software compatibility and increasing competition for those requirements MCF will not meet.

For example, the AN-UYK-19(V) is a computer built around a commercial instruction set architecture. There is only one producer. In 1977, a user survey revealed 34 versions of the same computer in 22 different developmental systems. The modules were not interchangeable nor were the chassis similar. The Army standardized the boxes and cards at a cost of approximately \$3.3 million and 15 man-years of effort. The investment has already shown a cost avoidance of approximately \$23 million in initial Integrated Logistic savings for the current nine users. The potential life cycle savings from the standardization effort is estimated to be in excess of \$100 million. The total program, which included a display, printers, etc., has a projected life cycle savings between \$100 million and \$300 million.

A second family of three computers with a different commercial instruction set architecture is being investigated for a similar effort. Initial estimates indicate a cost avoidance of \$8.1 million for an investment of \$7.3 million. A significant fact is the \$7 million dollar cost for the Integrated

FIGURE 2-2: Spares Investment for One Computer Family



SPARING ASSUMPTIONS

- **1,500 NEW MAIN FRAMES ON 250 SHIPS**
- **20,000 NEW MINI'S ON 500 SHIPS**
- **50% SPARING COSTS APPROX. ¼ COST OF ONE COMPUTER**

MINIMUM FLEET SPARING COST TO ACHIEVE A 0.97 OPERATIONAL AVAILABILITY WITH ONE COMPUTER FAMILY:

250 MAIN FRAMES	×	**$80K**	=	**$20 MILLION**
500 MINI'S	×	**$20K**	=	**$10 MILLION**
		TOTAL		**$30 MILLION**

Logistic Support package (less spares) for the three computers. Normal estimates run from $1 to $3 million for the LSAR data per computer, depending on its complexity.

Still, there is no direct competition between these computers, since the instruction set architectures are different. Each is essentially sole-source.

The Army has conducted numerous analyses and estimates in conjunction with the military computer family (MCF) program to estimate the potential for cost avoidance by reducing the number of computer types used from the current estimate of 60 to six types. Using data from a survey conducted in 1981 and extrapolating to an estimated 130 systems which will use computers in the 1990s, estimated quantities of 63,300 single-module computers, 21,600 AN/UYK-49s (microcomputers), and 15,030 AN/UYK-41s (minicomputers) will be in use throughout the tactical units of the Army. By surveying current systems, it was estimated that of the 60 types now planned to be in the inventory, there are 25 types equivalent to the MCF single-module computer, 15 types equivalent to the MCF micro, and 10 types equivalent to the MCF minicomputer.

It was then assumed that the transition period to the MCF was completed. (In practice, this would hot happen.) It was assumed that each computer, both MCF and non-MCF, would be kept in production for five years and would be in the field for 10 years. Reliabilities were assumed to be equal and based on those projected for MCF. Spares were considered to be stocked in normal locations within the Division. In one analysis, there was no learning curve data used, whicn means there was no average in production cost by producing larger quantities of one computer versus lower quantities of more types, and a second analysis using a 95% learning curve was used.

Given these assumptions, for a 20-year period, there would be a $592 million savings in spares and operational readiness floats alone, and with the 95% learning curve, that figure increases to $782 million. Naturally, with no learning curve, production cost would be the same between MCF and proliferation. But if 95% learning curve is used, there would be be over a billion dollar saving in production cost by reducing the types to that of the MCF program. These figures are somewhat hypothetical, but they indicate the direction the Army should take.

2.6 Integrated Logistics Support Effects from Reducing Unnecessary Proliferation

System development time is around seven to 10 years, and most weapon systems are designed to have a useful life of 15 to 20 years. Some remain longer and are upgraded over time. In the realm of electronics, and especially computer systems, semiconductor technology advances at a rate such that chips and computers are obsolescent within approximately four to eight years, and go out of production. The implication of this steady improvement in technology is that any proliferation reduction must maintain competition and include new technology. However, disregard for operational availability and integrated logistics support for technology's sake is expensive and risky. The two must be balanced. To counter this, a logical plan for preplanned product improvement and exploration of new technology must be part of the overall program.

The continued evolution of interface specifications is preferable to the continuous proliferation of new computer types.

The ISA specification is the interface between hardware and software. By maintaining this specification and external interfaces upwardly-compatible, new versions of software-compatible hardware can be available competitively across systems. There are a number of areas in which economies of scale can be realized by reducing the number of different computer types. There will necessarily be some different types to meet the various environments and size requirements.

2.6.1 Establishing Support

Beginning early in the acquisition cycle of a new weapon system, engineering drawings of system components and assemblies are analyzed to determine those parts that are candidates for repair or replacement. This analysis is a part of the LSA process.

With the proliferation of the many diversified computer systems, there has been an increased burden placed on all support documentation. The following areas have been significantly impacted by the proliferaton of computer types within DoD.

- Drawings and Specifications -- The number of drawings and specifications increases proportionally to the number of computer systems fielded by different manufacturers, as complete drawings and specifications must be maintained for all technical data packages. The proliferation of some 59 computer types in the Army currently in the inventory, has created a significant cataloging, storage and update problem for this documentation element. Reducing computer proliferation requires fewer sets of drawings and specifications to be maintained.

- Logistic Support Analysis and Records (LSA/LSAR) -- Proliferation of computer systems also increases the burden and expense of developing separate LSA/LSAR program for each computer system. LSA is by nature expensive and time consuming. However, if properly implemented, it results in extensive Life-Cycle Cost (LCC) avoidance by insuring that the system is supportable upon fielding. This process, however, is often slighted when development funds and time is critical, resulting in reduced supportability. With proliferation reduction, fewer LSA/LSAR efforts need to be implemented, which will result in a fully defined, efficient and economical support package at a considerably reduced cost than that presently being encountered.

- Technical Manuals (TMs)/Maintenance Allocation Charts (MAC) -- The requirement fcr TMs and MAC charts has increased dramatically as a result of the extensive proliferation of the computer system. This problem has increased even further when multiple manufacturers are introduced and form, fit and functions specification were used. The development, distribution, and update of the documents required under proliferation is a burden in itself. Reducing proliferation significantly decreases the magnitude of the TM/MAC problem, decreases publication costs and increases the probability of having current technical libraries in operating maintenance elements.

- Repair Parts and Special Tools List (RPSTL) -- Closely tied to the problems generated by technical manuals and MAC charts are the increased magnitude of repair parts and specialized tool list that are required when computers are proliferated. Each manufacturer currently must produce its own specialized repair parts and tools for each system and these must be fully documented for repair personnel. This impacts maintenance operating elements by significantly increasing the documentation (and, in addition, the parts and tools they must maintain). Reducing repair parts and special tools lists would greatly reduce these impacts.

As the system is fielded and the maintenance concept is implemented, system operation and maintenance is established. DoD's supply and maintenance procedures must account for rapid deployment and initial force operation until normal maintenance and resupply can be established for sustaining operations.

2.6.1.1 Training

Depending on the maintenance concept, training for new systems is accomplished by new equipment training teams for the equipment operators and maintainers. Later courses are set up in schools to provide replace- ment personnel to the units. The skills to be taught are derived for the LSA. To the extent maintenance procedures can be common, there are signifi- cant economies of scale to be achieved across systems that use common types of computers.

2.6.1.2 Stockage Requirements

It has already been noted that where practical, operational availability goes up and the number of spares to be procured is reduced by using common hardware. Each end-item, LRU, module or part to be a repair part must be individually packaged in a prescribed unit of issue. The development of packaging data sheets is based upon the physical characteristics of the item and the environments that it must be protected against during transportation, handling and storage, from the time it leaves a manufacturer's plant until ultimate use in the field. This is another major cost resulting from the severe operational environment. Resupply may be accomplished under battle conditions.

Currently, the development of packaging data sheets is repeated for each new component of computer subsystems under development. Even though there may be a great deal of commonality among components and packaging methods for similar items, a separate document is required for each component. Each document requires from two to six hours to prepare, and then must be processed, approved, and managed.

By reducing proliferation, redundancy can be eliminated. Once prepared, the same data sheets could be used by whatever contractor produces the computer subsystem or components to the approved military specifications.

2.6.1.3 Transportation, Handling and Storage

Movement of repair parts through the supply system is becoming an ever-increasing and costly problem. Because of their small size, electronic components create problems. Shipment from depots to requisitioning units can be delayed because components are packaged with other items destined for the same location. A partially-filled shipping box may be delayed at the depot until it is full enough to be closed and dispatched. Also, storage of small and low-density line items is a warehouse nightmare. Stocking the bins from incoming shipments and filling requisitions from stock is a labor-intensive continuous operation. Pilferage, miss-location storage and timing problems between receipt and zero balance requisition rejection are increased.

Reducing types of computer subsystems will reduce, but not eliminate these problems. By reducing the number of line items stocked and increasing the quantities of each because of the increased density in the field, inventory management, storage and accountablity would be improved.

In the field, each support activity maintains its repair parts in bins. Normal procedure is one part type per bin. Proliferation of computer types causes inefficiency in both volume and space utilization, as well as increasing the workload and potential error in inventory management and accountability.

2.6.1.4 Operational Readiness Float

Major commands are authorized to establish operational readiness floats (ORFs) in accordance with applicable regulations (AR 750-1 and AR-710-2 in the case of the Army) and to distribute them to support units for issue to using units. The purpose of any ORF is to meet prescribed operational readiness or availability rates and to improve the material readiness posture of assigned units. Support units issue ORF items to using units to replace unserviceable, economically repairable items that cannot be repaired or otherwise restored to a serviceable condition within prescribed maximum repair time limits.

The support units bear the burden of accountability, maintenance and storage of the ORF items. This responsibility is in addition to the requirements for maintaining an Authorized Stockage List and Direct Exchange items for each of the systems supported. A strong case can be made for reducing this burden by reducing proliferation of computer types. Instead of stocking ORF items for the multitude of different computer subsystems in varying quantities, based on hardware reliability and availability requirements, the support unit would stock a greatly reduced number and scope of ORF items. Accountability procedures would be less involved, due to the reduced number of different items. Maintenance operations could be streamlined because diagnostic and test equipment could be set up for a run of similar items that are unserviceable. Storage, inventory and retrieval operations would be simplified, because of the reduced types of ORF items on hand. Management of the ORF no longer focuses on many separate low-density line-items, each assuming crisis proportion when at zero balance. Rather, management could focus on serviceable versus unservicable quantites on hand, and schedule maintenance in light of current workload and ability to meet customer demand.

2.6.1.5 Test Equipment

Each system has test equipment, ranging from the equipment location itself, all the way back to the production line, if it exists. Newer equipment generally has some built-in-test (BIT) capability. There may be some functions in hardware, but usually there are diagnostics or software routines to isolate defective parts or circuit boards. As the hardware changes, so must the diagnostics change. After the operator uses diagnostics to locate a defective LRU, the LRU must be replaced with a spare and the defective components either discarded or repaired, depending on the cost. If the LRU is to be repaired, there is some further testing, maintenance and diagnostic equipment (TMDE) needed. If appropriate, and it usually is, some form of ATE (Automatic Test Equipment) is used at the appropriate locations.

TMDE and ATE is another area where gains can be made by reducing proliferation. Test Program Sets are software programs which run on ATE to test LRUs and circuit boards for repair, and again after the repairs are made, to insure the unit is functioning properly. Common hardware directly reduces the number of TPS programs and interconnect devices required to test the Units Under Testing (UUT). Nonetheless, software programs must be developed for each UUT. This is a major software management and distribution problem which can be reduced by reducing proliferation of hardware line-replacable units of circuit boards, modules and larger units where possible.

2.6.1.6 Software Support Centers

As the developed system is fielded, software support is normally turned over to the government. (In order to compete for contractor support or additional software, the support software should be vendor-independent, well understood and documented.) A particular problem is experienced in the software support centers when there are different software support systems. It allows proliferation of languages, compilers and other tools, along with host and target computers to be managed. Since for these real-time systems, the software is normally developed on a commercial host and transferred to the target system for operation, a set of the target hardware is essential to test the operational software. In many cases, simulators and other software must represent items of equipment which cannot be operational in the software support center. Transferring the data bases between host systems is expensive and time-consuming. For large systems, a separate center may be established for commonality. With other systems, a separate center is not too important except for training of programmers or when upgrades are accomplished. For smaller systems, a common software support environment permits supporting more than one system from the same host and allows new software tools, which can enhance productivity, to be developed for more than one system at the time. This allows for more efficiency of operations and programmer training.

Adoption of Ada will improve the problem, as far as languages are concerned, but Ada alone will not solve the host and support environment problem sufficiently for savings.

Significant savingscan be realized by a common environment and common language. If common ISAs are used for fielded computers, then savings can be realized in the software support tools that depend on the ISA of the target, such as linkers, debuggers, code-generator portions of compilers, loaders and any hardwire connections for down-line loading of the target computer.

2.7 Impact of Technology Insertion on ILS

During the life of a system, changes in the configuration of deployed hardware are necessary. The impact of this on follow-on support depends upon the ramifications on logistics--the existing support structures, tools and capabilities. The impact of configuration change must be evaluated during the review, analysis and approval process.

2.7.1 ILS Changes

The development, testing, production, distribution and application of Modification Work Orders (MWOs) is a scaled-down version of the ILS activities used in the system acquisition process. Many of the uncertainties associated with the deployment of a new system are removed when an existing system is modified. Planning is based on a known density of equipment, location and support infrastructure. Timing of application can be established and coordinated, based upon need. More accurate cost predictions can be developed for manpower, facilities, tools and equipment. Personnel training can be analyzed and developed.

Since changes to a hardware item may not be noticeable to the operator because the basic form, fit and function (F^3) are not usually altered, to the logistics community, hardware design changes are usually accompanied by changes in spare parts, special tools and equipment, and test and maintenance procedures. In addition, some form of training is usually required to apply the change. Then the actual process of implementing the change must be conducted, and finally, the supply system must be purged of the now obsolete items. This is routinely handled by Modification Work Orders (MWO) for specific pieces of equipment. The MWO can be completed at any level within the support system, depending upon the complexity, the technical expertise required, and the requirements for special tools, test equipment and facilities.

The MWO workload is in addition to the normal support operations for the system. The density of the system and the extent of the MWO will dictate the duration of the additional workload.

Although it is unlikely that present computer subsystems would require an MWO at the same time, with proliferation reduction, this possibility becomes realistic. Resolution of the problem depends upon the degree of planning, urgency and resources allocated. Savings in the areas of training, facilities and production methods can be achieved in large scale modification efforts. Also, the engineering required to develop the change is less because the number of computer types will be reduced.

Presently, any repairman of the multitude of computer subsystems is expected to be a "jack of all trades". Existing automatic test equipment has been stretched to the limit of its capability. Technology advances will continue to require development of elaborate interface modules between new hardware and old test equipment. Right now, test equipment operators spend more time setting up for a test routine than isolating problems in unservicable items.

Reduction of computer types is a step in the right direction. The range and scope of repair parts would be drastically reduced. Personnel training could focus on fewer subsystems, thereby increasing proficiency. By consolidating the existing maintenance personnel, dedicated support organizations could be created.

2.7.2 Configuration Management

Most systems discussed here are large, real-time systems which are centrally developed and managed, and then distributed to military users and units all over the world. Such things as differences in flight control from one fighter to another cannot be tolerated, nor can tactical fire direction systems differ between units. Training would be practically impossible, and personnel could not be reassigned to different units.

Changes to an item's specifications, drawings or production baseline design is normally accomplished through Government-controlled configuration board, consisting of representatives from subsystem users, support organizations, material developers, readiness commands and industry. Recommended changes are presented in engineering change proposals and evaluated in terms of overall impact upon the item life-cycle cost and impact on operational availability, logistics support, producibility and quality assurance.

With reduction of proliferation, fewer configuration control boards would be necessary to evaluate the impact of each proposed change.

Plans to ensure the viability of logistic support could be developed. The necessary documentation, training and support elements could be developed, tested and introduced into the field along with with the change in configuration or application of the Modification Work Order (MWO). For some systems, a change may be only in software, but this change may affect the unit, operator, or equipment so greatly that it amounts to almost a new system.

2.8 Commercial Production Lines

Currently, the commercial computer marketplace is highly competitive, led by rapid strides in electronic technology. While the basic technology and configuration remain for the life of the production line, manufacturers are continually making improvements, normally reflected as model changes, to their products. A commercial microprocessor being manufactured today will most likely be supplanted by a new model four years from now and essentially "forgotten" within 15 years. It normally takes nearly 15 years to develop a weapons system that requires an embedded computer. The normal service life is 15 to 20 years. It is necessary to retain software which has been developed and tested over hardware upgrades, plan for upwardly-compatible ISAs for the life of the system, or make special provisions to obtain replacement parts for those that go out of production.

Coupled with the "short" life of commercial computers is the problem of proprietary data rights. Many operating procedures, component designs and manufacturing techniques are proprietary in nature. This often makes it expensive or impossible to purchase requisite technical data (for a second-source production contract) or to transition maintenance procedures to military depot maintenance operations.

When purchasing a computer from a commercial vendor, the Government essentially has no say in the configuration. The vendor is able to (and often does) change configurations during a production run in order to meet his commercial requirements. This significantly complicates the military support of the item, as two (or more) models are in the field, each with different repair parts, technical documentation, and test equipment requirements. The Government also has no control over the component configuration of the vendor-produced model, and must accept what is assembled, even though components which do not provide optimum operation in the military environment are used.

Software compatability has already been discussed in some detail. Due to the current policy of using a proliferation of commercial computers, there is fielded equipment that uses some 10 HOLs and 30 machine languages. While this may not be a problem when viewed from the individual system level, it does become significant when we look "backward" at the support echelons required for maintenance of both the software and hardware.

Note that in any major electronic device for military application, the cost of the operating electronic components will constitute a very small portion of the cost of the end-item itself. Even in the "worst" cases, the cost of only the operating components will be a small portion of the end-item cost. Therefore, even if there are significant cost increases in procuring MIL-STD electronic components, this additional cost will be quickly offset in increased maintenance and provisioning costs and, more importantly, reliability improvements experienced will significantly increase operational readiness.

2.9 Ensuring Repair Part Availability

The rapid technological changes experienced and projected for computers has and will continue to shorten the active life of computers. Consequently, the provisioning of spares and repair parts will be comparatively low compared to end-item density.

Manufacture of spares after last end-item production run may not be undertaken, as this consumes manufacturing capability that can more profitably be used in manufacturing the latest technology and items and their associated spares.

The most economical method of maintaining fielded computer systems is not necessarily the method that provides the operational availability required.

The problem is further complicated by the cost and operational availability trade-offs between "buyout" and maintaining a production capability to provide repair parts and spares for the projected service life of the system. This "buyout" permits purchase of spares and repair parts at lower unit cost. There are inherent risks with "buyout" such as:

- Actual consumption rate of repair parts and spares could be higher than projected.
- The service life of the system could be extended beyond that projected.
- There could be latent defects in the system or individual components which could require re-engineering, replacement and subsequent repair part support.
- Modifications and product improvements cannot be freely undertaken if there is an impact on spares or repair parts which cannot be recaptured.

There is also the cost of storage and inventory management associated with maintaining a large inventory over a period of many years as a result of "buyout".

The alternative to buyout is, of course, maintaining a repair parts manufacturing capability by either paying an exorbitant price for parts which are obsolete by commercial standards and provided by manufacturer only because it is so profitable, or by directly subsidizing to a manufacturer to maintain a production capability and retain qualified vendors. Either option is costly.

The current proliferation of military computers poses a serious logistic support problem as well as seriously effecting operational availability.

Reducing the number of computer types will obviously reduce the number of parts that must be considered for retention in production or "buyout". By reducing the number of types and planning for "preplanned product improvements" (either entirely new version or improvement to current versions), the risk of having to keep old technology in production or "buyout" spare parts is reduced. This also provides the alternatives to move to a software-compatible newer version.

SECTION 3 TECHNOLOGY OVERVIEW

3.1 General

Two of the major areas of technology which have emerged since WWII and have led to tremendous advances in weapon system capability are computer science and solid-state electronics. The power and flexibility of programmable data processors, combined with the small size, low cost and high reliability of microelectronics, have revolutionized military systems. It is now possible to put more computing power on a handful of circuit cards than the total throughput of all the computers which were installed in the U.S. in the year the integrated circuit was invented (1957). Radars (command control, communications and intelligence (C^3I)), electronic warfare (EW), missile guidance and many other system types require highly-competent computers to perform their missions and to remain able to deal with evolving threats.

In recent years, the ability of complex microcircuits to implement an entire computer with a handful of components has led to widespread use of "embedded" computers. These processors are components of a larger system and may well be invisible to the user. They bring the flexibility of programming to relatively low-level system functions. However, the proliferation of embedded processors and associated software makes it imperative that good management and design be employed to control costs and ensure reliable operation.

Computer science, which deals with system architectures, software, performance measures, and solid-state technology, which furnishes the means to implement complex digital machines, continue to make progress at explosive rates. Today's computers are smaller, far more reliable, and obviously more powerful than those of a decade ago. It is vital to the free world that this technology be rapidly exploited in military systems. The remainder of this section outlines the principal solid-state components, processor categories and software issues involved in military embedded computers.

3.2 Solid State Technology

The transistor was invented in 1947 and the first integrated circuits (ICs), which place multiple inter-connected transistors and other components in a single package, appeared in the late 1950s. The components have grown dramatically in speed and complexity. The following paragraphs summarize the major device types of interest for embedded computers.

3.2.1 Logic ICs

The earliest commercial IC units contained a dozen or so transistors. Commercial parts now in production contain up to several hundred thousand transistors on chips not much larger than the first ICs. Modern parts are also much faster, so that their net processing capacity is much greater. In some applications, all this throughput is not needed to meet basic system requirements and can be used to support other features such as reliability and reduced software cost. Since the size of a producible chip is upper limited (to about 0.3" square), the 10,000-fold increase in circuit complexity has resulted chiefly from a shrinking of individual element dimensions from 10-20 microns (um) in the late 1960s-vintage chips to 3-5 um or less in the latest generation of ICs.

Both bipolar transistors and metal-oxide semiconductor field-effect transistors (MOSFETs) can be used as the switches from which logic is assembled. Table 3-1 summarizes the main logic families currently in commercial production and their characteristics. It is conventional to speak of four generations of IC complexity: small, medium, large and very large-scale integration (SSI, MSI, LSI, and VLSI). The LSI era dawned in 1970 with the introduction of the first complete programmable data processor on a single chip (called a microprocessor), and the first large memory chip (holding 1024 bits, i.e., 1K bit). With VLSI, what would once have represented a complete digital system can be built on a chip. Moreover, as the level of integration increases, the cost per circuit element drops, and for a given number of circuits, system reliability improves due to the reduction in external interconnections. Since power dissipation per circuit element also decreases with size, the VLSI chip still only consumes 0.5 to 2.5 watts. Chip complexity is limited by such factors as the number of input/output leads available in packages, the ability to design and test highly complex circuits, and the maximum chip size which can be produced with acceptable yield (fraction of total chips on two wafers which are functional after processing). Development programs such as the DoD Very High Speed Integrated Circuits (VHSIC) program and other VLSI efforts in industry have lead to steady improvements in the characteristics shown in Table 3-1.

At VSLI levels of complexity, traditional manual design methods are too slow and costly. Three general types of computer-aided design (CAD) systems have evolved. In a fully custom environment, computer graphics aids the designer in drawing chip patterns, storing the design, and checking for compliance with rules. In a library-driven scheme, smaller circuit functions called "cells" are developed and stored in the computer; special software then places cells selected by the designer on a chip and works out the connections. Finally, in "semicustom" logic design, a wafer full of general-purpose logic elements is "personalized" by developing an interconnection pattern which "wires" the chip as a particular circuit.

3.2.1.1 Microprocessors

Three general microprocessor (uP) types have evolved in commercial practice. The most common has a Central Processor Unit (CPU) on a single chip. This is supplemented by memory, input/output (I/O), and other chips to build the complete computer. A variation on this, often called a "microcomputer" or "microcontroller", combines CPU, memory and I/O on a single chip, usually at some penalty in sophistication or performance, but with obvious cost and size advantages. The third type and most exotic uP type is the "bit-slice", in which multiple CPU chips are paralleled to build up a processor of arbitrary size. Bit-slice designs are usually both the most complex and the highest performance in the uP world.

TABLE 3-1: DIGITAL INTEGRATED CIRCUIT FAMILIES
(In Commercial Production in 1982)

Type	Gate Delay (ns)	Power Dissi- pation (mW/gate)	Layout Density (gates per sq mm)	Radiation Hardness
I. Bipolar				
A. Transistor-Transistor Logic (TTL)				
(1) Low-Power Schottky (LPS)	10	2	20-80	Can be made high
(2) Advanced LPS	1-2	1-2	200-400	
B. Emitter-Coupled Logic (ECL), including variants such as Current Mode Logic (CML)				
(1) Standard ECL	0.5-1	25	20-50	High
(2) LSI ECL	0.7-1	10	100-200	High
C. Integrated Injection Logic (I^2L)	2-5	0.05	200-400	High
II. Metal Oxide Seimconductors (MOS)				
A. P-Channel MOS (PMOS)	50	1	100-200	Low
B. N-Channel MOS (NMOS)				
(1) 5 um Channel	10-20	1.5	200-400	Can be made moderately hard
(2) 2 um Channel	1-2	0.5	1000-2000	
C. Complementary MOS (CMOS)				
(1) Standard	20-50	0.01-0.1	50-100	Can be made high
(2) LSI CMOS	1	0.01-0.5	500-1000	
(3) CMOS Silicon-On-Sapphire (SOS)	0.5-1	0.01-0.5	750-1500	
D. Exotic NMOS Structures (vertical, double-diffused, etc.)	0.5-1	0.5	400-800	Low

General purpose uPs are available in most of the logic types of Table 3-1. Word lengths vary from four to 32 bits, memory addressing as high as several megabytes, and speeds up to about 10 million primitive operations per second are commercially available. In general, CPU chips are members of a "family" of compatible memory, I/O, control and other support components to simplify the assembly of a complete computer. Single chip microcomputers are very widely used in 4-bit and 8-bit versions, executing a few hundred thousand operations per second. Special versions are available which include functions such as analog-to-digital conversion, or a counter/timing circuit. The last category, bit-slice parts, are built in two, four and eight bit widths, usually supported with speed enhancement, such as pipeline registers and character generation. They are most often used in very high throughput applications, such as mainframe computers and digital communication controllers. Almost all use bipolar logic types for high speed.

3.2.1.2 Auxiliary Chips

One great benefit of LSI/VLSI to the system designer has been the integration of complex support functions on single chips. Parallel and serial I/O ports, Direct Memory Access (DMA) controllers, interrupt priority chips, counter/timers, and many other functions replacing dozens of SSI/MSI parts and eliminating much of the design task are available. An increasingly popular support chip is the "coprocessor". This works with the CPU as a fast special-purpose calculator for such things as trigonometric functions or floating-point arithmetic. In applications using extensive complex calculations, this hardware implementation can run far faster than the alternative, which is to use software routines in the general-purpose CPU.

3.2.1.3 Semicustom Logic

Many manufacturers supply gate arrays, programmable logic arrays and similar components which initially are completely general in function and are then customized by the manufacturer or user. Gate arays are personalized as described in an earlier paragraph. Arrays of up to several thousand gates (equivalent to mid-1970s uP) are available today, and 10 times more complex arrays are expected within two years. These components allow the "random logic" portions of a system to be efficiently achieved with a few chips. This contrasts with the early days of LSI/VLSI, when the size advantages of the main chips were largely lost by the need for several dozen SSI/MSI components in supporting roles.

3.2.2 Memory

Semiconductor (i.e., integrated circuit) memory has largely supplanted the magnetic core memory which was the mainstay of computer systems up to the mid-1970s. The two general memory chip types are RAM (which stands for "Random Access Memory" but by some quirk of terminology, actually implies "read/write memory"), and ROM (which stands for "Read Only Memory"). Variants of the ROM are the PROM (programmable ROM), whose contents can be stored by the user and the EPROM (erasable PROM), which can have its contents wiped out by an ultraviolet light or voltage signal, allowing reuse. ROM can only store information which never changes or changes only over long intervals because erasure is impossible in some cases, and time-consuming in others. It is, however, non-volatile, meaning that the contents are retained even if the power supply shuts down. RAM, by contrast, is general-purpose, but volatile. In most uP systems, ROM is used for software and constant storage and RAM for data and such software that changes over time. Several approaches to a true non-volatile RAM, i.e., one which has the same speed and unlimited read/write cycle life-time as volatile RAM, are being pursued. Within five years, such components should be available. A further important distinction is between dynamic and static RAM. The former uses a single transistor per stored information bit and thus packs more memory on a given chip than static RAM, which uses four to six transistors. However, dynamic RAM, requires "refresh", i.e., every location must be examined and rewritten every few milliseconds, while static RAM does not. Refresh wastes about 5% of total memory access time and makes the chips both harder to use and vulnerable to the transient effects of nuclear radiation.

ROMs and dymanic RAMs are routinely available at capacities up to 64 Kbits, although not commonly usable over the full military temperature range. Speed, which is usually measured by "access time", the time from interrogation of the chip until contents are available for reading, varies from several hundred nsec per large chips, down to 100 nsec or less for parts in the 16 Kbit range, and as little as five nsec for emitter-coupled logic (ECL) parts holding a few bits. Parts at the 256 Kbit level are beginning to appear, and 1 Mbit chips will be available in five years or less.

3.2.3 Advanced and Exotic Components

3.2.3.1 Compound Semiconductors

Virtually all logic ICs are built on silicon chips, since this is a mature, low-cost and easy-to-manufacture technology. Compound semiconductors such as gallium arsenide and indium phosphide offer speeds 5-10 times greater than silicon. They also may someday permit integration of microwave and optical elements on the same chip with logic. Also, these materials are intrinsically far more tolerant of nuclear radiation and high operating temperatures. The first truly commercial gallium arsenide logic components will appear shortly. Experimental chips at MSI to LSI complexity have been demonstrated. While the lower cost of silicon will always be preferred when its performance is adequate, compound semiconductors will complement silicon in very high data-rate applications and where radiation hardness or integration with optical and microwave signals are important.

3.2.3.2 Josephson Junctions

For ultrahigh throughput machines, where the limit on speed is the velocity of light and the packing density of the logic elements, Josephson Junctions may be the fastest logic yet devised. Operating at liquid helium temperature and dissipating miniscule amounts of power, they, in principle, allow a large mainframe computer to be packed in a few cubic inches. Josephson Junction machines are years away from being fully developed, but may one day play a role in such high throughput applications as C^3I information centers.

3.2.3.3 Magnetic Bubbles

Much closer to practical use in military systems, bubble devices use a tiny magnetized region in a thin film on a garnet chip to store millions of bits per square centimeter. Although they are serial-access devices and intrinsically much slower than RAM or ROM, they are non-volatile. They are thus attractive for uses like those served by disk drives in conventional computer systems, i.e., mass storage of software and data files not needed in main memory. With very high bit-density, low cost per bit, and no moving parts, bubbles are well-suited to military systems. However, work remains to be done to extend their operation to the full military temperature range. The commercial market has been slow in developing, so the potentially very low cost of mass-produced bubble memories has not yet been realized.

3.2.4 VHSIC

The Very High Speed Integrated Circuits (VHSIC) program is developing two generations of VLSI components tailored to the system needs and operating environments of the military. By establishing pilot production lines and demonstrating chips in brass boards, the program stresses access to VSLI by military users. Systems such as automatic target recognizers and anti-submarine warfare acoustic signal processors can generate digital signal processing throughputs of billions of operations per second, achievable only with VSLI. VHSIC also addresses supporting CAD tools, component reliability and testability and a wide range of standardization issues. These standards range from basic implementation standards for package types, power supply voltages, etc., all the way up to architecture and instruction set standards. VHSIC will make the benefits of VLSI available to military users far faster and more directly than would be the case in a "business as usual" interaction with a commercially-oriented industry.

3.2.5 Summary and Forecast

During the 1970s, IC complexity roughly doubled every year. Clock frequencies, the basic measure of logic speed, increased from a few hundred kilohertz in the first uPs to 10MHz or more in the fastest contemporary VSLI parts. Even if the rates of increase slow, million-transistor chips and 50-100 MHz clock rates can be expected within a few years. Steady progress is also being made in logic families which limit power dissipation and in ways to increase radiation hardness. The tremendous processing power of these chips and the storage capacity of modern memories mean that the performance limits of many systems will be set by software and by the task of managing complexity in VSLI production capacity, virtually any system architecture will be buildable in a compact, reliable machine.

3.3 Processing System Architectures

From the invention of the computer until quite recently, virtually all data processing systems consisted of a CPU supported by an array of I/O and data storage peripherals. Advances in computing power resulted mainly from faster CPUs and peripherals, larger memories, more efficient software, etc. In some cases, multiple computers were used under control of an executive program, which allocated available resources among tasks. In addition, networks of geographically-separated computers have become commonplace to transmit information and share storage and computing cpacity.

The advent of small, powerful processors is revolutionizing the architectural approach to many applications. Distributed processing networks can achieve both high overall throughput and fault tolerance through redundancy. Such systems pose difficult issues of functional partitioning, control and synchronization, and software design and verification. Properly managed, such systems promise flexibility, performance, and reliability not possible with one or a few computers. Several classes of multiprocessor architectures are becoming familiar.

3.4 Instruction Set Architecture

3.4.1 The Role of ISA Standards

Standardization is possible at virtually all levels of a computer system. As with any standards process, the goal is to achieve the economic and reliability payoffs of standards without unduly restricting design flexibility or freezing in obsolescence. Given the rapid evolution of digital circuitry, standards at the detailed hardware design level would be too short-lived to be useful.

Historically, the two most successful classes of standards have been programming languages, especially higher-order languages (HOLs) such as FORTRAN, COBOL and Ada, and instruction set architectures (ISAs). In principle, though never perfectly in practice, HOLs allow transportability and reuse of software on different computer models. They also support improved programming efficiency, better software verification and easier modification of existing programs.

At the other end of the software development spectrum, an ISA defines a set of machine commands, representing the elements of "object code" into which the HOL "source code" is translated for loading and execution. One of the most famous and imitated ISAs is that of the IBM/370, itself a descendent and refinement of previous ISA generations. If strictly adhered to, an ISA supports software transportability even better than an HOL, since it avoids the notorious idiosyncracies of compilers (very large and complex programs which translate HOL source code), and it reduces expensive diagnostic testing of software such as that written specifically for aircraft and missiles as well as space systems. It allows technological advancement in the computer itself, since faster, more efficient implementations of the ISA and more reliable components can always be introduced. It also permits refinement of programming environments, since the input (the HOL) and the output (the ISA) of a language system are defined, and therefore, the translation process itself is free to evolve.

In a rational standardization scheme, both HOL and ISA standards contribute to an environment where competition is encouraged, while costs are controlled and total system reliability is enhanced. The DoD approach to computer standards, using both HOLs, such as JOVIAL J-73B and its eventual successor, Ada, and ISAs such as MIL-STDs 1750A and 1862, is just such a balanced system.

In any computer, basic machine functions are defined by an instruction set. This is composed of "macroinstructions" such as ADD, SHIFT, COMPARE, etc. By the nature of digital circuitry, each such instruction is executed as a series of "microinstructions" which enable the appropriate circuit elements at the proper time. A "microprogram" performs the vital control task of translating the machine language macroinstructions of the executable program into the signals which actually enable the electronics. In effect, the microprogram defines the instruction set since it determines which commands will be executed correctly. The consequence is that a given ISA can be implemented on a wide variety of hardware architectures by appropriate microprogramming.

In recent years, computer designers have increasingly tended to use microinstructions stored in memory vs. the older scheme in which these control actions were defined by the pattern of a complex switching circuit ("hardwired" control). Fast, dense, memory chips, together with advances in CPU hardware design and high-speed components, make this approach attractive from many standpoints, including design cost and verification of design correctness. It also supports ISA standardization by simplifying the effort involved in marrying the ISA to a particular circuit technology and supporting subsequent changes as that technology matures and improves.

To be fully useful as a military standard, an ISA must efficiently execute the very wide range of computational tasks encountered in military systems. Most commercial ISAs are optimized for specific applications (data processing, machine control, etc.). In developing ISAs such as MIL-STD-1750A and 1862, extensive efforts have been made to consult the widest possible spectrum of expert opinion and to carefully balance features of the architecture for wide applicability. If the result is not "optimum" for any specific class of algorithms, when compared to the advantages of standardization this not significant. Moreover, the high throughputs possible with VLSI make up for any residual inefficency in most practical cases.

3.4.2 Advanced and Exotic ISAs

Architectures such as the MIL-STD-1750A are certainly not the end of the evolutionary path. Computer scientists are experimenting with new concepts which, if proven feasible and attractive, may become the basis for future machines. Two examples are the HOL direct execution machine and the minimal instruction set machine.

A direct execution (DE) machine would eliminate all run-time software except the original HOL source code as written by the programmer. The commands of the HOL would be executed by invoking microcoded routines stored in memory in the control unit of the computer. For obvious reasons, this scheme is very attractive, and experimental DE machines can be expected to appear in the near future. As a partial example, one commercial 32-bit microprocessor executes selected Ada commands via routines in on-chip ROM.

However, a true DE architecture for modern HOLs can be extremely complex, requiring both a variety of special-purpose circuits and a very large control storeage. Opinion is divided among experts in the field as to whether the DE approach is preferable to spending the same effort on optimizing more conventional processors and associated software tools. An example illustrates the difficulty: One of the important features of modern HOLs is strong data typing, in which the exact nature of stored information is explicitly declared in each program module using that data. In addition to making the resulting source code easier to read and verify, strong typing guards against errors involving illegal or inappropriate operations. In a compiled environment, the complex process of checking for violation of type rules is done once, at compilation. A DE machine would have to do some degree of type checking every time data is accessed, or else submit the program to a rules check, in effect a partial compilation, before trying to run it.

Experiments are also underway to define architectures with minimal instruction sets. A prominent example is the University of California (Berkeley) Reduced Instruction Set Computer (RISC). The idea is to design a highly regular hardware set which efficiently executes a streamlined instruction repertoire containing the most-used functions of a chosen application. Payoffs include very high throughput on programs which are well-matched to the instruction set, reduced design time, and reduced complexity, especially in the control hardware. How well a machine of this type can execute the diverse computational tasks required of a standard military computer must be determined by further work, but the basic premise is promising.

3.5 Summary

The combination of advanced architectural concepts, such as distributed processing networks for r^liable real-time control, and advanced solid-state technology, which allows such systems to be built, is a potent tool for maintaining the qualitative superiority of free-world weapons. This capability must be harnessed to military needs in a fashion which controls costs and development times, while ensuring reliability and flexibility. Comprehensive standardization-embracing HOLs, ISAs, physical and electrical characteristics, interfaces, etc, are a key element in managing and exploiting this capability. The tremendous progress in microelectronics makes it possible to achieve the benefits of standardization without unacceptable penalties in system complexity, cost and modernization.

SECTION 4 ANALYSIS OF ALTERNATIVE APPROACHES

4.1 The Standardization Issue

In Section 1, problems related to DoD's prior approaches to the use of embedded computers were discussed. These problems are:

(1) Expensive and lengthy system development and evolution.
(2) High cost of computer system software.
(3) Inability to reuse/transport applications and support software.
(4) High cost of logistics and maintenance support.
(5) Inability to acquire, train, and retain competent military field maintenance personnel.
(6) Low operational availability (survivability, failure and casualty backup, continuity of operations).

In this chapter, the factors that have caused these problems are identified and various solution approaches are explored.

4.2 High Order Language (HOL)

It is generally recognized that extensive use of assembly language in the development of software for weapons systems will result in higher software costs, longer system development time, more latent software errors, and more difficult and error prone future modification than would be the case had any good high order language (HOL) been used. The basis of this conclusion is that a software program in assembly language is generally much more complex, more difficult to understand, and much longer in terms of the size of its source program (as much as ten times longer) than its equivalent in HOL form. In recognition of this, all of the Services initiated use of HOLs in the development of software for embedded computer systems.

The early uses of HOLs have tended to be reasonably successful in the short-term. These earlier HOL's often required minor to substantial sections of programs to be in assembly language because of the state-of-the-art of HOL's in the late 1960's to mid-1970's. As HOL's were introduced, the extensive subsequent proliferation of different HOLs created long-term complications as well as compounding the inevitable embedded assembly language use. One such long-term complication is the inability to reuse/transport software to a replacing next generation computer which is optimum for the upgraded requirement and which employs an ISA and/or HOL different from the one originally used to develop the software. Because of this, competition for the next generation computer was found to be non-existent or very limited. Also, the Services found themselves using different and incompatible compilers, software tools, interfaces, peripherals, etc. This situation was complicated further as most of the HOL systems in use were not highly developed, not well documented and not competently supported. Furthermore, the limited experience for each different HOL and its associated environment required programmers to be retrained each time they were required to work on a system that employed a different HOL and environment.

In 1974, OSD (Office of the Secretary of Defense) concluded that HOL standardization was essential and that standardization on any good HOL would be better than no standardization at all. A joint OSD-Services working group established a set of requirements for a standard HOL and it became clear that none of the existing languages satis- fied the requirements well enough to become a long-term DoD-wide standard. At the time, the Services' HOL programs were uncoordinated with multiple dialects, incompatible compilers, and different host environments. Several of these HOLs were inextricably linked to the ISA of a particular computer. With such motivation the Ada effort started[1,2,3,4]. (A detailed description of the Ada Program can be found in Section 5).

HOL standardization can be expected to reduce the extent of assembly language programming, eliminate the need for retraining, improve the quality of embedded computer software, increase the likelihood of achieving transportability, reduce the cost of software development and post-deployment support, and reduce the time required for system development and evolution. HOL standardization is considered to be essential--a necessary tool. However, as will be discussed in subsequent sections, it is not sufficient. It will not be a panacea with respect to DoD's software problems. Two other equally important factors in the software equation are the programming support environment (PSE) and the instruction set architectures (ISA) of military computers. Also not addressed by HOL standardization and discussed in subsequent sections are the problems of logistics/maintenance support and operational availability/survivability.

The benefits to be derived from HOL standardization will diminish over time in a relative sense if the standard is frozen for too long a period. If the long-term standardization policy is to be effective it must contain plans for the management of change. The standard should be permitted to evolve in increments, each of which should permit (unless impractical), software that has been previously developed using the standard HOL to continue to be recognized as being the evolving language. While the evolving standard may have a long life--25 or more years--long-range plans should address the advances that will be made in HOL technology to insure that a successor standard will be established when appropriate.

4.3 Programming Support Environment (PSE)

Software for an embedded computer system is by no means fully represented by its collection of HOL and assembly language programs. Consideration must also be given to the relationship between the HOL software and the software tools used in its development and maintenance. Any practical software system must be viewed as a living/dynamic entity that changes over time as errors are experienced through use and as changes are made to accommodate an upgrade system, mission or equipment. The total system that is used for software development and support is called the programming support environment (PSE). Included as integral parts of the PSE are the following: The HOL and its compiler; the various levels of specification and documentation (much of which may be automated and dependent upon a specific support computer); configuration control aids; software tools such as editors, performance monitors, debugging aids, preprocessors, tracers, verifiers, etc.; software used to communicate with the host operating system which are in the form of an HOL-like command language; the host computer (including its ISA, operating system, and physical interfaces to operators, programmers, and military computers); and the military computer (including its ISA, operating system and other software products, and its physical interfaces).

The portion of the PSE that extends beyond the HOL has had, and will continue to have, a significant influence, positive or negative, on the DoD software problem. The all-too-frequent and uncontrolled practice of system contractors using different and unfamiliar HOLs, PSEs, ISAs, and support computers in system development incurs high costs and schedule risks. The same holds with respect to DoD's support elements. For example, one of the Army's eleven post-deployment software support centers, which has responsibility to support fielded battlefield communications systems, is burdened with the necessity of using and supporting 15 different high-order languages, 30 different instruction set architectures, 28 different types of host computers and 39 different types of battlefield computers. When a change must be made to a different programming support environment, even if a standard HOL is used, all of the above must be taken into account. Thus, standards covering the programming support environment are considered as important as those for the HOL. Without such standards, system contractors and DoD software support centers will remain overburdened stemming from a myriad of different and incompatible programming support environments. This will result in the inability to reuse/transport applications software with respect to new military computers and the inability to reuse/transport support software with respect to new host computers as well as new military computers, even if a standard HOL were used universally.

4.4 Instruction Set Architecture

4.4.1 Need for Reuse/Transportability of Software

During the lives of embedded computer systems, existing computers have had to be replaced for a variety of reasons. To meet changes in the threat addressed by the system or to satisfy a new system need, existing computers have had to be replaced by units that had greater computing speed and greater memory and input-output capabilities. In some cases, size, weight, power, reliability, or environmental performance of the current computers had been unsatisfactory. Computers also had to be replaced because they were obsolete and their logistics and maintenance support costs had become prohibitively expensive. In many of these cases, much, if not all, of the original software would have remained valid for the upgraded system, but instead had to be redone, i.e., starting with redesign and development of a new software essentially from scratch, was required because it was deemed more efficient than trying to translate at the end-product level. The basis for this impediment to transportability was the fact that new computers had different HOLs in their programming support environments or incorporated because of different ISAs in their designs. Thus, many of the computer approaches to an upgraded system that appeared to be better technically or that offered lower hardware acquisition costs than other altenatives could not be pursued due to the added time and expense that were required to redo the software. This dilemma could not be resolved through the use of a new computer that supported the original HOL but did not incorporate the original ISA, due to the strong dependence of the original software on the original replaced ISA. This dependency, which is a critical aspect of ISA standardization, is discussed in Section 4.4.2. Clearly, if a replacement computer were available that employed the original HOL, PSE, and instruction-set architecture (ISA), then the transportability of existing software would be feasible and a significant amount of redevelopment avoided.

4.4.2 Dependence of Software on ISA

In this section it will be shown that: (1) ISA dependencies are pervasive in the HOL, PSE, and in run-time software products (such as a real-time operating system); (2) ISA dependencies are necessary in current practice and reflect the state-of-the-art of both hardware and software; and (3) ISA dependencies are an impediment to effective reuse/transportability of software.

The first part of the problem is that the software system of an embedded computer system can rarely be expressed entirely in HOL, without assembly language*. Direct use of the ISA (rather than the HOL) is required to develop certain portions of the software such as:

(1) Parts of the operating system,
(2) Input-output programs,
(3) Time-critical functions for which the program produced by the HOL compiler does not meet the required system response time,
(4) Functions for which the program produced by the HOL compiler requires large amounts of memory such that the computer's memory capacity is exceeded, and
(5) Features not available in the HOL.

The second major part of the problem is that HOL programs themselves are also ISA-dependent. Dependence of HOL software on the ISA for which it was developed exists with respect to all high order languages (including Ada) used in practice to develop systems. A program can produce radically different meanings (computations) in systems with different ISAs.

HOL programs can be ISA-dependent in fairly obvious ways; classic examples are the dependence on representation specifications and unchecked conversions. Minor differences in floating point arithmetic instruction in the ISA can affect termination as well as accuracy of the HOL program. HOL programs also can be ISA dependent in far more subtle ways. One dependency derives from the natural tendency of programmers to write HOL programs that would be efficient for the underlying features of an ISA which in no way guarantees efficiency of the same program for other ISAs.

In fact, an order-of-magnitude reduction in efficiency may result. Such software tends to involve pervasive data structure and algorithm choices to the extent that the transfer to a new ISA would require a redesign as well as a rewrite. As another example, fault-tolerant programs tend to be ISA depen-

* The instruction set architecture and the assembly language of a given computer are essentially identical. Thus, assembly language software is entirely dependent on ISA. It is, in fact, "ISA software".

dent, in that neither the expected errors nor the appropriate recovery actions would be the same for different ISAs. This is critical in such applications as flight control.

Ada has been designed with the goal of minimizing ISA dependencies to the maximum extent feasible. As a result, Ada has fewer ISA dependencies than most other languages and likely the least for any language of its capabilities. With the current state-of-the-art limitations, however, many dependencies remain, such as [5,6,7]:

(1) range of integers and associated error conditions,
(2) alignment and layout of integers,
(3) range, precision, rounding, and handling of overflow with respect to floating point numbers,
(4) low-level input-output,
(5) range of priorities,
(6) duration of clock timer intervals,
(7) interrupt handling and interrupt entry points,
(8) task switching and swapping,
(9) representation specifications,
(10) derived types and overload,
(11) address specifications, and
(12) exceptions.

In summary, while the use of the HOL is clearly preferred over that of assembly language, the current state-of-the-art, unfortunately, does not permit exclusive use of HOLs in embedded computer system software. Furthermore, it cannot be predicted at this time when, if ever, such dependencies will be completely eliminated. Thus, a significant and critical portion of both the embedded software and the support software can be expected to have to be redone, at considerable time and expense, if an existing computer were to be replaced with one having a different ISA.

4.4.3 Battlefield Dependence on ISA

In addition to the software basis for ISA standardization in Section 4.4.2, there is also an operational basis which is summarized below and more fully explored in Section 4.5.

As discussed in Section 1, the high cost and complexity of logistics and maintenance support of a large number of different types of embedded computers and the requirement for casualty/failure backup of critical systems during battle have become serious concerns. In Section 4.5 these concerns are addressed and the conclusion reached that controlled and evolutionary standardization with respect to battlefield and shipboard embedded computers is essential. This need then cannot be met, practically, without each such embedded computer incorporating, if not a standard, at least a de-facto ISA.

Furthermore, the need established in Sections 1 and 4.5 for a capability of practical on-line tasking of other embedded computers that are located within the same battlefield environment to restore some (or all) of the functions of a damaged/failed computer must also be addressed. (Of course, such a backup capability would have to be incorporated into the original design of each system.) A simple and cost effective solution to achieving this type of "field interchangeability" of battlefield/shipboard software function is to use computers employing the same ISA; the alternative, a costly way, is to conduct multiple software developments, one for each of the different ISAs to be employed in the backup scheme. The latter alternative is the result of computer proliferation.

4.4.4 Competition and the Protection of ISAs

As discussed in Section 4.4.2, software reuse/transportability can be achieved through the replacement of an obsolescent computer with a new one having the same HOL and ISA*.

With the existing widespread proliferation of embedded computer ISAs, however, the probability is high that replacement computers with the same ISA as the computer to be replaced could only be acquired from a single industry source, a procurement approach that violates acquisition policy and is not in the best interest of DoD. This situation can be anticipated since all of the ISAs in use in DoD's embedded computers, with the exception of those that are part of existing standardization programs, are not appropriate for industry-wide competitive procurements since they are protected by patents, copyrights, or trade secrets. (The same holds for almost all commercial ISAs, i.e., ISAs used in off-the-shelf commercial computers.)

Various protection mechanisms (patents, copyrights, and trade secrets) are employed by companies to retain the economic value of their ISA. An ISA has value primarily on the basis of a marketable, competitive product-line of computers and support software associated with it. Companies protect their computers and support software but not for the same reason they protect their ISA. A computer is protected to prevent another company from copying it. A support software product is also protected, not only because its development was expensive, but, unlike the computer, it is very easy to copy. Once copied, such support software, particularly that which is valuable to the community-at-large, would lose the economic value it has to its developer, since the latter would no longer be able to profit monetarily from it. Even if the computer and its support software are protected, there is still a need to protect the underlying ISA. Another company could use that ISA as the basis to develop a competing computer, of a different design, that would be capable of executing the same applications and support software.

Traditionally, ISAs have been used as the basis, not of a single computer, but of an entire product line produced as a family by a single company with a ready-made, exclusive market for its next generation computer, namely the users who own a current version and would like to upgrade to a more powerful (or more cost-effective) computer without having to change the software. is important to note that such upgrades, historically, have been made through the use of a company-standard ISA, the level of computer structure at which software transportability can be effected with minimal or zero change.

* Problems may still exist if, as is so often the case, the HOL or the ISA offered with a new computer is a subset of or an expanded or slightly different version of the HOL or ISA used originally.

In practice, DoD has found that the number of sources of a replacement computer with the same ISA has been limited usually to one--the manufacturer of the original computer. Such a procurement is sole-source and can be expected frequently to result in a higher costs and a poorer technical approach. Currently, throughout DoD, there is widespread lock-in to particular companies and their product lines, to the government's detriment. The situation has been complicated further where the original supplier of the ISA failed to have new ISA-compatible products in the line which met, without starting over, the specific needs of the DoD system for which upgrading was essential to meet a threat.

Essentially, then, ISA dependencies in computers and HOLs have placed DoD in a "Catch-22" situation: Moving to a different ISA will cause severe problems; staying with the same ISA will also cause a different set of severe problems.

4.4.5 Government Licensing of Commercial ISA

Given that commercial ISAs are protected, hence proprietary, would it make any sense for the government to gain the right, through purchase or license agreement, to use a limited number of such ISAs (limited for the reasons discussed in Sections 4.4.2, 4.4.3. and 4.4.4) in competitive acquisitions of mission-critical computers? Such an action would allegedly provide a source of commercial computers having the standard ISA (and a complement of support software) that, while not militarized and not suitable for general field use, could nevertheless be useful immediately in the development of weapons systems.

During the course of this commitment, the commercial owner of the ISA would, contrary to policy, receive continuing sole-source awards for their line of commercial computers having the selected ISA for government use in software development and support, use in benign environments, etc. The owner of the ISA would benefit further through continuing sole-source sale/license to the government of ISA-dependent support and system software. The owner might also receive royalty payments on a long-term basis from the government. Other benefits would accrue to the owner through the publicity of being chosen by the government as having a superior ISA, etc. In addition to the above, being contrary to acquisition policy, DoD could be expected to dissipate much non-productive energy dealing with the remainder of industry concerning their future ISAs, computers, and software products. Thus, standardization on a limited number of otherwise suitable commercial ISAs will result in a form of sole-source lock-in that seems to have as many unavoidable pitfalls as those resulting from the uncontrolled use of proprietary ISAs discussed in Sections 4.4.2, 4.4.3. and 4.4.4.

4.4.6 Use of Government-Owned ISAs

The remaining alternative of standardization on government-owned ISAs is next explored. Experience has shown that use of government-owned ISAs can achieve highly-effective open competition for military computers and associated software. This approach has the disadvantage that, initially, there would be a lack of computers having the standard ISA for use in software and system development. Another initial disadvantage is the possible unavailability of an adequate support software base. However, once government programs build momentum, these needs will be satisfied. Witness the Navy's AN/UYK-7 and AN/UYK-20 standardization efforts. Witness also the government-owned Air Force 16-bit standard ISA Program which has led to multiple competing sources for computers having the ISA specified in MIL-STD-1750A and to a complete set of JOVIAL language-based support software. The joint Army and Air Force standard 32-bit ISA standard known popularly as Nebula and officially as MIL-STD-1862, now has under development an Ada compiler, a comprehensive programming support environment based on the use of one or more types of commercial computers, and a multi-level secure real-time operating system.

Efforts to insure that DoD ISAs are state-of-the-art and of high quality are detailed with the presentation of individual Service programs in Section 5.

4.5 Embedded Computers

4.5.1 Analysis of the Effects of Computer Proliferation

Many of the problems now being experienced DoD-wide are directly attributable to the proliferation of different and incompatible embedded computers; expensive and lengthy system development and evolution; high cost of logistics and maintenance support; inability to acquire, train, and retain competent field maintenance personnel; high cost of maintenance training; and low operational availability (survivability, failure/casualty backup, continuity of operations). When system contractors are required to employ computers which are unfamiliar to them in system development, inappropriately high costs are incurred and schedule risks are increased.

4.5.1.1 Logistics and Maintenance Support

The situation with respect to cost of logistics and maintenance support of computers in fielded systems, as discussed in Section 1, is now very serious. Factors contributing to this cost include the development and maintenance of an Integrated Logistics Support (ILS) System for each different embedded computer type deployed, reliability testing, inventory management, test equipment, tools, ATE software, maintenance training, operational readiness float, and initial and replenishment spares and repair parts. ILS development alone typically costs $3 to $5 million per computer type and takes two to three years to complete. Yearly maintenance of the ILS package is approximately $250,000 for each different type. Even though the number of "in-system" fielded computers is the same with or without proliferation, the volume of spares, repair parts, and floats required is increased considerably since a minimum level of spares, parts, and floats is required for each unique type.

The fundamental basis can be illustrated by an example. If it were required that ten spare boards be kept on hand for each instance of a unique embedded computer in the field, then twenty boards would be required if two computers existed at the same location and each was unique, forty boards for four computers, each being unique, etc. On the other hand, if, in the four computer case, all four computers were identical, then a total of only 10 boards would be required. This number would increase to 11 boards for 10 identical computers vs. 100 boards for 10 uniquely different computers.

This argument also applies to the number of required military field maintenance personnel. DoD is very much concerned with being able to provide a sufficient number of military personnel capable of being trained to perform maintenance of sophisticated computers even if the number of different computer types in the inventory were small. With the proliferation of types now extant in the Army, for example, it appears that the present and anticipated needs for use of embedded computers to meet the prevailing threat will be thwarted due to lack of ability to maintain such equipment. Thus, unbridled proliferation quietly erodes DoD's maintenance capability.

Proliferation of computer types also has a serious impact on the cost of training, as discussed in Section 2.

4.5.1.2 Operational Availability/Survivability

The extensive use of computers has raised the level of sophistication of military systems to the extent that it has virtually eliminated manual backup in the event of battlefield-induced equipment breakdown. This has raised concerns in DoD about the ability of military field systems to survive, to continue functioning, and to be restored to operability in the face of computer outage due to component failure or battle damage. If computer damage is extensive, then a high density of common computer types will improve the likelihood that spares, parts, floats, and maintenance personnel will be available in the vicinity to restore systems to normal operation.

Alternatively, with a low density of each type, if supplies are exhausted and supply lines are cut, then the failure of even one system may critically render entire battle units inoperable. In such situations, proliferation of types seriously removes the possibility of taking like computers that are operating properly, but not performing key battle functions at a given point in time, and substituting them into those critical systems that have suffered computer damage/failure. Proliferation of computer types thus can become a major impediment to survivability and continuity of operations.

Also, the lack of commonality would impede reaching a level of continuous operation more sophisticated than that achievable via cannibalization, specifically, the effective transferability of software to other computers in the field. This "resumption of function" could not otherwise be carried out simply and straightforwardly, if at all. (Providing for this capability also requires planning in the original design of a system-of-systems network that provides for "graceful degradation" of function rather than catastrophic failure.)

Generic hardware proliferation problems will not be "resolved away" as technology advances, i.e., as computers become smaller and more reliable. Most of the problem will remain; few will become less severe. Advancing technology alone will not solve the problems of logistics/maintenance support and survivability (operational availability) of future embedded computers.

4.5.2 Mission-Critical Computer Standardization Alternatives

Preceding sections explored in detail problems caused by the proliferation of DoD mission-critical computers and concluded that controlled and evolutionary HOL standardization and ISA standardization were necessary but would provide only a partial solution. They would not reduce the high cost of field logistics and maintenance support nor would they eliminate problems associated with maintenance training. They would not improve operational availability and they would only provide partial relief of the problem of expensive and lengthy system development and evolution. Without a suitable approach to embedded computer hardware standardization, DoD anticipates continuing difficulties with respect to the cost, survivability, supportability, and growth of field systems that contain embedded computers.

In addressing alternative approaches to hardware standardization, consideration must be given to technology, competition, service operating and support environments, system growth, and cost.

Computer technology has been advancing very rapidly. Computer speed and circuit density, for example, have been doubling approximately every three years, and this impressive rate can be expected to continue. Concomitant increases in reliability and decreases in size, weight, power, and cost/performance ratio can also be expected to continue. In the analysis of alternative approaches to standardization, therefore, consideration must be given to computer technology in light of the potential for rapid obsolescence of embedded computers from threat, cost, and logistic support viewpoints. (The advance of hardware technology is considered to be much greater than that of HOL and ISA technology to the extent that it makes sense to think of multiple generations of hardware implementing the same ISA and having software supported by the same HOL.)

With standardization on a limited number of government-owned ISAs, service hardware acquisitions can be accomplished on a fully competitive basis since lock-in to a single company based on its protected ISA could not occur. If well-structured, such acquisitions should be met with an ample and enthusiastic industry response, witness the 12 bids by 15 participating organizations for the advanced development of the Army's Military Computer Family, and also the 23 companies that have demonstrated in-house developments and now stand ready to bid Air Force solicitations involving MIL-STD-1750A ISA computers.

Several hardware standardization alternatives warrant consideration. The level at which standardization is approached in each can have a significant bearing on how technology is dealt with, on how competition is employed, on logistics support, on operational availability (survivability), on life-cycle costs and on system evolution. Each alternative will be discussed keeping these factors in focus. The number and types of standard products and the duration of use of such products are also variables that will be considered. Not considered at this point is the applicability of each standardization approach to individual Services and whether a single standardization program should apply to more than one Service. That discussion is in Section 5.

4.5.2.1 Total-Product Standardization

The first level considered is standardization on the entire product (its design, components, and physical form) such that instances of it would be identical, inside and out, with the exception that some capabilities such as memory and input-output, may not be at the maximum level or capacity provided for in the product design. Several types of such products may constitute the standardization approach of a particular Service (e.g., the Navy's AN/UYK-43 and AN/UYK-44 and the Army's UYK-41 and UYK-49 computers). Such an approach provides the least burden in terms of logistics and maintenance support and provides for high operational availability and system survivability potential. Provision could be made for each such product type to be available from multiple suppliers with acquisitions on a competitive basis. Other than for second-source development, the internal hardware design of the standard computer would not be pre-specified by the government, but would be expected to change with each new generation of the standard (i.e., every four to eight years). Thus, the life of each product would tend to be limited by:

(1) The rate of technology which might have a negative effect on the availability of repair parts, and

(2) The need for more advanced products in order to meet new enemy threats. This approach, while providing the highest degree of standardization raises several questions:

   (a) to what extent would the government be locked-in to the supplier (in the single producer case)?
   (b) If multiple suppliers of identical items were established, would competition result in substantive benefit to the government? (i.e., would competition be "meaningful"?)
   (c) How much of a risk would the government be taking with respect to technical performance due to the production of a single design for Service-wide use? (i.e., what is the risk that the wrong technical approach might be chosen?)

4.5.2.2 Form-Fit-Function Standardization of Computer Modules

The second level is standardization on a form, fit, and function basis with respect to the computer's internal plug-in modules and the physical form and external interfaces of the computer. In such an approach, each module type would be acquired competitively using a form-fit-function specification. Modules of a given type made by several manufacturers would be required to operate in place of each other but would not be required to employ the same components, technology nor the same internal design. This approach is not as good as the first level of logistics and maintenance support because of different piece parts. This approach also involves the high risk that DoD or a third party cannot develop and manage effectively form-fit-function specifications at this level that are complete, consistent, unambiguous, and independent of technology. The high speed internal bases of a computer, in fact, are technology-dependent and, traditionally, controlled by the computer manufacturer, not the government. Also form-fit-function partitioning itself tends to be highly dependent on the particular type and level of technology employed by the computer manufacturer.

4.5.2.3 "Box-Level" Standardization

The third approach, "box-level" standardization (form-fit-function standardization of the computer "box") entails specification only with respect to the external attributes of the computer. This level of specification, which is less dependent on technology than module standardization (Section 4.5.2.2), usually covers ISA, physical form, performance, capacities, environmental requirements, and external electrical and human interfaces. Provision could be made for each product type to be available from multiple suppliers, each employing his own internal design and technologies, with acquisitions on a competitive basis. (For example, two designs may be fielded with two suppliers for each.)

In contrast with the first and second levels discussed above, this approach offers a compromise solution. It reduces the logistics and maintenance support burden but not nearly as much as the first two alternatives. (The three approaches would provide varying reductions of the currently prevailing high state of computer proliferation). This approach also provides more options with respect to technology, provides for a broader technology base for DoD embedded computers, and offers a more extensive basis for competition. It also provides the potential ability to use interchangeable units of like type produced by different manufacturers. This approach does require that interchangability of like types (having different designs) in the field would have to be validated during development.

4.5.2.4 Standardization on a Limited Number of External Attributes

The fourth approach is a relaxed form of the third level of standardization in which the specification of external attributes would be limited to ISA, environmental requirements, and perhaps some external electrical interfaces (i.e., full form-fit-function specification would not be employed). A limited number of such products would be permitted in the inventory; system developers would be required to limit competition to existing suppliers, once an adequate number of suppliers has been developed. In general, units could not be used interchangeably.

4.5.2.5 Standardization Only on ISA

The fifth approach is to standardize on ISA only and to implement hardware control on a system-by-system or fleet basis. This approach may be suitable for those systems whose hardware may have to be specially configured, or whose computational needs may not be met via existing standard hardware products. A fleet approach may be suitable in those instances where a large number of the same type of hardware items would be employed and where logistics support wasnot common beyond systems within the fleet. In this case, DoD would still be able to derive the benefits described earlier with respect to the use of HOL and ISA standardization. This approach goes the furthest of all toward broadening competition and facilitating technological insertion.

4.6 Summary

4.6.1 Need for Reuse/Transportability of Software

During the deployment lifetime of weapons systems, embedded computers are replaced for a variety of reasons:

(1) To meet changes in the threat addressed by the system.
(2) To improve efficiency or accommodate system enhancements.
(3) To replace obsolete computers that are too costly and unreliable to keep in the system.

In many cases, much of the original software may be valid for the upgraded system and savings would result through the ability to use this software in the new computer, thereby avoiding the expense of a major redevelopment.

4.6.2 Inadequacy of HOL Commonality

Reuse (or transportability) of software is effectively prevented if the new computer does not exhibit the original ISA. ISA dependencies are pervasive in the HOL, the programming support environment, and in software products such as a run-time operating systems. ISA dependencies are necessary in current practice and reflect the state-of-the-art of software and hardware design. Dependence of HOL software on the ISA for which it was developed exists with respect to all HOLs intended for system development. In other words, a HOL program will frequently result in radically different computations by computers with different ISAs. Also, order-of-magnitude differences in efficiency can result in the execution of an HOL program on computers of comparable speed, but with different ISAs.

Ada has fewer ISA dependencies than most other languages. However, at least twelve major areas of ISA dependencies have been identified. It cannot be predicted at this time when, if ever, ISA dependencies will be completely eliminated. Thus, a considerable portion of the software can be expected to have to be redone if an existing computer were to be replaced with one having a different ISA. Therefore, software transportability is dependent on ISA commonality.

SECTION 5 SELECTED DOD STANDARDIZATION APPROACHES

5.1 Introduction

During the past decade, the Department of Defense, (DoD) and the Services have embarked on a number of computer-related programs to manage computer resources effectively and to reduce proliferation in those areas where operational availability and tactical force capability are foremost. These efforts address both software and hardware issues, while maintaining focus on unique Service and mission requirements. The DoD common-language project, Ada, instruction set architeccures (ISAs) such as MIL-STD-1750A and MIL-STD-1862, as well as service standard hardware programs such as the Army's Military Computer Family (MCF) project and the Navy's AN/UYK-43/44 projects have come from these standardization programs. These efforts will allow DoD to field cost-effective systems, while maintaining control over major factors causing geometric growth in computer-related costs.

Although there are many elements of DoD's standardization program, the basic thrust is to standardize consistent with technology and mission requirements, life-cycle support issues, and competitive procurement practices. Because of these factors, DoD and the three services have implemented several standardization approaches which include Ada, MIL-STD-1862, MIL-STD-1750A and several standard hardware procurement programs. These standards are vendor-independent for competition and can be improved over time.

This section of the study will describe the various DoD and Service programs that are underway.

5.2 Ada Joint Program

Ada is a modern high-order computer programming language which will become the standard language for writing software for DoD mission-critical computer applications. The Ada Program extends well beyond simple language configuration management and will help control the cost and improve the quality of software by facilitating the application of modern software development practices. The Ada Joint Program Office (AJPO), attached to the Deputy Under-Secretary of Defense for Research and Advanced Technology (DUSD(R&AT)), is managing the DoD effort to introduce, implement, and provide life-cycle support for Ada.

5.2.1 Background

It has been estimated that the DoD currently spends about $4 billion per year on embedded computer software. This includes the design, development, acquisition, management, and operational support and maintenance of such software. The role of software in embedded computer systems is expanding, counter numerical superiority. But this technical superiority exacts a price--increasing complexity. The basic software problem is related to the software problem is related to the rapidly growing complexity of systems coupled with existing software practices. Recognizing the need to improve software business practices, DoD started an initiative to focus management resources on the improvement of software life-cycle practices and tools used in DoD's computer systems.

In 1975, the High Order Language Working Group (HOLWG) was established--with representatives from the Army, Navy, Air Force, Defense Communication Agency (DCA), National Security Agency (NSA), and Defense Advance Research Project Agency (DARPA)--to investigate the feasibility of adopting a common high-order computer programming language for use on embedded computer systems.

A comprehensive set of requirements was developed through extensive coordination in DoD, allied countries, industry and academia. Over 26 existing computer languages were formally evaluated against these requirements. No existing language was sufficient to serve as the common language. The HOLWG undertook a competitive international procurement for the design of a language to meet those requirements. Funds for this activity were provided by the Services and technical management was provided by DARPA. An initial design of the language was completed in May 1979.

5.2.2 Language Standardization

The language design team, CII Honeywell Bull, completed the language design and the government published the Language Reference Manual (LRM) in July, 1980. On December 10, 1980, the reference manual was republished by the DoD as a military standard (MIL-STD-1815). Since that time the AJPO has been working towards making the Ada Language Reference Manual a universally-recognized specification.

The AJPO has used the canvass process as the procedure for establishing Ada as an American National Standards Institute (ANSI) standard. The canvass procedure began in April 1981 when ANSI approved the canvassee list and the canvass package, including the July 1980 LRM. A ballot was mailed to the list of 96 organizations representing potential implementors, potential users, and general interest categories.

The ANSI standardization process came to a successful conclusion on February 17, 1983 when Ada was voted an ANSI standard. An ANSI Ada reference manual is now available. Efforts in the future will be directed toward the adoption of Ada as an international standard. Standardization will provide stability for the development of a comprehensive software environment and reusable software.

The European software community, which has participated actively in the language design and reviews, recognizes Ada as a state-of-the-art computer language and is very supportive of the Ada program. The European Economic Community (EEC) has adopted Ada as its common implementation language. The NATO Community is considering Ada as a standard to facilitate interoperability of systems and to reduce NATO life-cycle embedded computer costs. NATO's Military Command and Control Information System Working Group (MCCISWG) established an ad hoc group of experts to consider the implications of adopting Ada as the standard for NATO funded portions of NATO command and control information systems.

5.2.3 Ada Programming Support Environment (APSE) Development

The Ada program involves more than just a language. By design, it incorporates many features needed to support modern software engineering practices. An intrinsic principle of modern software engineering is the use of an automated environment to provide complete life-cycle software support. Recognizing the importance of environments, it is the goal of DoD to develop and maintain all Ada software on state-of-the-art Ada Program Support Environments (APSE) which are based on the "Stoneman" model ("Stoneman" is a requirements document for support environments).

The purpose of an APSE is to support the development and maintenance of application software throughout its life-cycle, with particular emphasis on software for embedded computer applications. One of the more important concepts in an APSE is the database, which acts as the central repository for information associated with each project throughout the life cycle. The database supports the organizational infrastructure, as well as maintaining the data critical to the management, development, testing, and life-cycle support of software. The database also serves as the interface through which the set of modular tools communicate and interact with each other. The database will contain management information such as version control, library support and project management as well as the code, test data, and documentation required as part of any life-cycle support.

The DoD has two Ada Programming Support Environment developments under way, the Army Ada Language System (ALS) and the Air Force Ada Integrated Environment (AIE). It is the Navy's intention to provide a standard support environment based on the Army ALS and a corresponding run-time environment to meet the demands imposed by specific mission-critical requirements. These efforts are for Minimal Ada Programming Support Environments (MAPSEs) because the AIE, the ALS, and the Navy ALS will initially comprise only a little more than the "minimal" tool set required for life-cycle support of software. Consistent with the Stoneman model, additional tools can and will be developed and added to these environments.

The Army, with Navy and Air Force participation, has begun full integration testing of the ALS in 1983, and when satisfied with the ALS operation, it will be made available for rehosting. It is planned that the ALS will be fully available for production software development and maintenance in 1985. The ALS is initially hosted on Digital Equipment Corporation's VAX 11/780 VMS, but is designed to be rehostable. Additional tools will be added when requirements are determined. The AIE is scheduled for initial delivery to the Air Force in 1984. The AIE is hosted initially on the IBM 370. The AIE and the ALS will initially reside on different host computer hardware and provide alternative design approaches which reduce overall program risk. Funding two developments ensures availability of a reliable support environment at the earliest feasible date. Eventually, the DoD will determine the best approach to APSE design and the Service's APSEs will converge to a DoD standard.

5.2.4 Insuring Tool Transportability

It is reasonable to assume that additional tools will be recommended for inclusion into the Ada programming support environment portfolio in the future. It is important to be able to absorb these potential proposals in an orderly manner so as to converge to a DoD-wide shareable APSE.

In the nearer terms, it is necessary to coordinate the AIE and ALS development efforts to converge to a standard set of host computer/APSE interfaces in order to provide a basis for considering additional tools. The host computer/ASPE interfaces reside in an inner core software implementation referred to as the Kernal Ada Programming Support Environment (KAPSE). The KAPSE contains the host computer unique hardware and software primitive implementation parameters peculiar to a specific computer. To the MAPSE, the KAPSE provides the standard interface so that the MAPSE design is independent of the internal KAPSE design. This independence provides the basis for APSE tool portability.

To focus on the convergence of the ALS and AIE MAPSE/KAPSE developments, the Navy has been selected to lead a joint service review team to identify and recommend conventions within the KAPSE. The review team has been named the KAPSE Interface Team (KIT). An associated volunteer international group of both industry/academic experts has been established to support the KIT.

The payoff of this effort is to consolidate the portability gains made by the Army ALS and Air Force AIE development efforts and to entertain evolutionary additions to the APSE definition. Success of this effort will assure Ada tool portability which, in turn, will provide savings to DoD.

5.2.5 Ada Run-time Considerations

The real-time performance of a system is a combination of both the supporting hardware and the language in which the system was written. Although Ada supports writing efficient real-time programs, it depends on the machine's specific code generators, optimizers, and run-time systems to turn that potential into reality.

By concentrating attention on selected standard ISAs in the near term, more attention can be devoted to the development of very efficient code optimizers and run-time systems. The new ISAs proposed for military standardization do provide facilities for efficient execution of real-time programs. Efficient Ada software, which can take full advantage of these ISAs, is not available. During the near term, DoD needs to focus attention on the development of code generators and optimizers for designated standard instruction sets. Commercial usability of Ada is an important feature in overall language standardization and use; but DoD has particular needs of its own. Funding the development of efficient run-time systems is needed because of their distinctive character compared to commercial systems. For example, consider the optimization of the space needed for program storage or computational resources. In a commercial environment, memory is a relatively cheap commodity, therefore there is little incentive in today's hardware marketplace to develop space-efficient programs. On the other hand, a military system is usually extremely large in the number of options, and therefore, memory is relatively expensive. Memory is more expensive because of DoD requirements for volatility, ruggedization, radiation hardening, power, weight, and operational availability. This means that military systems will constantly be facing space-optimization problems which are almost irrevelant in the commercial community. These needs will not be stressed by the developers of commercial Ada developers of commercial Ada language processors. Without some way of focusing attention on particular ISAs, it will be impossible to devote the resources necessary to develop the efficient code generators, optimizers, and run-time systems required.

Ada has a logical relationship to computer ISAs. The degree to which an ISA provides support to Ada Language constructs has an economic impact. The closer the architecture of an ISA matches the Ada Language, the greater the probability achieving economic Ada application code portability. The Army and Air Force have been focusing on this issue since 1979 and have evolved the MIL-STD-1862 Nebula ISA to closely support the Ada Language.

The Ada/MIL-STD-1862 combination will give birth to the practical portability of applications software from one weapons systems to another. This is the primary means to support interoperability among systems and functional survivability within a system from one generation to another.

5.2.6 Transportability with Ada

Ada provides considerable opportunities for transporting software from one program to another. But these are still just opportunities. Effective programs for software transportability include not only the high-level programming language, but the ISA of the eventual target computer, the run-time system on that target computer and the surrounding support software. Even with standardized ISAs, there are many other areas in the run-time system and supporting libraries which will need standardization to achieve full code transportability. Use of Ada as a high-level programming language is one step in solving the computer system's problems. ISA standardization is another step. Together these can have an effective impact on improving the use of computer hardware and software. Without the ISA standardization, DoD can expect only limited benefits from Ada in the near term.

The Ada Program involves a number of features, but there are still hardware dependencies. Particular attention has been placed on the development of a standardized computer programming language. Through the use of a standard programming language, it should be possible to transfer high-level programs, transfer personnel, reuse training resources, and reuse support software tools. These activities are all undeveloped at this stage. The actual transporting of code from one program to another is only a small part of the effectiveness of the overall Ada program. Even if all that was possible was to transfer development support software, and to provide a common language for the development staff's communication with other human beings, Ada would be a significant accomplishment. Through the use of common ISAs, DoD will be much more likely to be able to reuse applications code. This means that the Ada program by itself can achieve considerable improvements in the development of DoD software. when combined with ISA standardization developing with Ada, it has an even larger potential for cost control and improved technology.

It has been called to the attention of many people that a large U.S. semiconductor company has developed an ISA closely related to Ada. This should not be blown out of proportion. There are a number of ISA and hardware architecture proposals that are closely related to the Ada language. Some of these provide tremendous opportunities for efficient execution of Ada software. There is, however, also the view that low-level architectures provide more flexibility in the translation of programs and therefore can be more effective in the overall execution of programs written in a high-level language. At this state it would be improper to drop DoD standardization efforts with conventional architectures and move completely toward any specific high-level architecture directly related to Ada. Considering the ways programs are developed, it is very likely that the more conventional architectures will, in the near term, be the most effective method of executing programs. As a further note on the Ada-related instruction set architecture, the overall architecture has been much less than satisfactory in its execution speed of programs. This should not be taken as a negative reflection on Ada, as much as it is an indication that the development of this kind of new instruction set architecture is very difficult and it may take a long time to reach the execution capabilities we have in the existing standardized ISA's.

Ada can be used to write reusable code. Reusable code can be viewed in two different ways. First, reusable codes can transcend one generation of specific weapon systems to its successor. Second, reusable code can be inserted into two different systems to support interoperations standards and to distribute identical processing capability to two or more hierarchical systems which perform the same function.

Additionally, reusability also means that the Ada encapsulated mission function can be used on dissimilar ISAs without reinvention of the function solution. The form of the software solution will follow the mission function to preserve the solution without incurring complete redevelopment cost. Today, software solutions are uniquely "part of" a system solution. With Ada, many mission functions will be separable and "used with" a system or systems solutions.

It should be noted that only the DoD-sponsored ALS and AIE developments are supportive to providing a rehosting capability and multiple code generators to more than one ISA. These two specific goals are over and above the commercial developments now in progress. It is the successful implementation of these goals which provides the leverage for the anticipated cost savings within DoD.

5.2.7 Compiler Validation

Enforcement of a single definition of Ada as a standard is the only method available which will ensure portability of embedded computer software at a reasonable cost. This can only be achieved if every compiler conforms to the Ada standard. The DoD has taken two steps to enforce the Ada standard. Trademarking the term "Ada" and developing a capability to certify Ada compilers. The DoD will not validate Ada subset, superset, or language variant compilers for the development or maintenance of DoD Ada software. An Ada Validation Office (AVO), reporting to the Technical Director, AJPO, will ensure that Ada compilers implement the standard independent of hardware manufacturer.

Historically, a compiler implementation has been a de facto definition for the language. Consequently, when different compiler developers make divergent implementation decisions, dialects appear. The goal of this Ada program is to provide non-divergent implementations of Ada. Consistent with the goal, the purpose of the Ada Compiler and KAPSE tool interface validations is to ensure the maximum transportability, interoperability and reusability of Ada software at the least cost to the users and ultimately the public. This will be accomplished by maintaining a firm Ada standard represented by the Ada Language Reference Manual, publishing implementation guidelines, and providing a powerful compiler validation capability. Since there is virtually no chance that two compilers will implement precisely the same language without some mechanical check, the AJPO has developed a sophisticated validation capability which will test compiler conformance. There will be no restriction on the distribution of the validation test set. This is beneficial, not only because it will assure conformance, but it will be an enormously useful tool for implementers by providing a considerable measure of confidence in the correctness of the implementation.

The Ada Implementors Guide is available to all interested parties. It defines test objectives and discusses the ramifications of critical sections of the Language Reference Manual. Nearly 1400 test objectives have been documented in the Implementors Guide and over 1600 test programs have been written and tested. The complete set of approximately 1600 test programs has been upgraded to reflect the language changes made during ANSI standardization. The AJPO will continue improving the state-of-the-art in compiler validation through the Ada test set. Other validation tests may be developed independently where practical, and will be used to augment the Ada Validation Office's test suite. Anyone may submit a test for inclusion in the test suite to the Ada Validation office, which will evaluate the merits of adding the proposed test.

Validation is vital to the success of the Ada program and in controlling the cost and improving the quality of software. The AJPO's policy is to minimize the compiler development effort required for validation. Validation guidelines have been published and are available through the National Technical Information Service. The guidelines can be summarized as follows: A party submitting a compiler for validation will be responsible for any direct costs incurred by the AVO during validation testing. The testing will be at the site of the compiler developer. Compiler developers whose compilers successfully pass all the tests will be issued a validation certificate which will be effective for one year. In order to assure continued conformance, annual validations of compilers will be required.

5.2.8 Introduction of Ada

The strategy for introducing Ada will vary with the individual Service and Agency needs and capabilities. Although there is a long-term goal to adopt Ada as the common language for mission-critical computer applications, some components have a current commitment to other languages and support systems. Depending on the stability and sophistication of those systems within a DoD component, components' strategies for introducing Ada will differ. Two prerequisites to Ada's introduction include the existence of an adequately tested APSE or commercial equivalent and the development of an organizational support infrastructure, which will monitor: The configuration control, distribution and maintenance of Ada software; education and training of DoD personnel;

industrial management; and the establishment of policies which promote modern software engineering practices. Each service must obtain and commit the resources required to develop the necessary support for Ada.

In general, initial introduction of the language will focus on new systems rather than modifications of existing systems. Programs coded in some other language should not undergo a translation unless they are undergoing a planned redesign for economic or performance advantage.

Several commercial efforts have been announced, including compilers by Western Digital, ROLM, TeleSoft, and Intel. All four companies have announced plans to market complete, validated compilers. To date, the ROLM and Western Digital compilers have been validated. It is anticipated that most major computer manufacturers will develop Ada compilers.

All existing commercial compilers are incomplete with respect to DoD sponsored developments in that they do not support rehostability nor do they currently generate code for DoD-selected computers. It is doubtful that private capital will be available to extend their use for DoD projects. In addition, as these compilers are new and the ACVC tests do not include compiler performance measures nor object code efficiency measures, it is highly likely that DoD use of these early compilers would necessarily require DoD funding to improve their utility.

5.3 Joint Army And Air Force MIL-STD-1862 Program

5.3.1 Background

In 1975, an in-house, ad hoc committee, called the Computer Family Architecture Committee, was formed to conduct comparative analysis of military and commercial ISAs and to recommend a single ISA for use as a standard for embedded computers. The Committee's recommendation, which was based on a variety of factors, was that the ISA employed in the PDP-11 product line of Digital Equipment Corporation (Digital) be selected[8]. Based upon this recommendation, the Army initiated negotiations with Digital in 1977 to obtain a license to use this commercial ISA.

By mid-1979 the Army had been negotiating with Digital for over two years to obtain a license for use of the PDP-11 16-bit ISA in its Military Computer Family (MCF). Negotiations also addressed future licensing of the new 32-bit ISA incorporated in Digital's VAX-11/780 computer*. The ISAs from those two computer programs would give the DoD both 16-bit and 32-bit ISA standards.

* The Navy did not join the Army in this effort due to the decision to replace the AN/UYK-7 and AN/UYK-20 with new software-compatible computers and to postpone introduction of a new ISA to the following generation of embedded computers.

In 1978, the Army engaged C. S. Draper Laboratory to perform an independent study of the MCF Project. One of the conclusions of this study was that future Army requirements for computers would be satisfied more effectively through use of a single ISA with 32-bit (vs. 16-bit) capabilities, and that the ISA of choice, in light of the negotiations that were ongoing, was the VAX-11/780[9]. The use of a single ISA for the entire family would provide software compatibility among all battlefield computers which offered the significant benefits of reduced support software costs, flexibility for system growth, and battlefield failure/casualty backup (survivability).

The Army accepted this recommendation but had become very concerned about the feasibility of obtaining a satisfactory agreement with Digital. There were also fears that close ties to a single commercial company via this ISA agreement might have an impact on the openness of hardware competition and hence on the success of the MCF Program. During deliberations on these issues, the Army decided to pursue in parallel a backup approach--use of a government-owned 32-bit ISA. Since none of the existing government-owned ISAs provided a true 32-bit addressing capability, the Army's choices for implementing the backup approach were either to modify an existing government-owned ISA in order to achieve the desired 32-bit structure or to design a new 32-bit ISA. After some trial design work, it was concluded that none of the basic structures of existing government-owned ISAs permitted satisfactory expansion to a full 32-bit capacity. Thus, in September 1979, the design of a new 32-bit ISA was initiated. Requirements for this ISA, later called Nebula, were as follows:

(1) Government-owned (vendor-independent).
(2) State-of-the-art.
(3) Efficient for military real-time systems.
(4) HOL-oriented.
(5) Optimized for Ada.
(6) 32-bit ISA including 32 bits of virtual address (to support up to 4.6 billion bytes of memory).
(7) Suitability for high speed implementations.
(8) Accommodation of inexpensive, low-performance members in the family having the full ISA.
(9) Suitability for multi-level security.
(10) Compatibility with future advances in VLSI technology.

It became clear to the Army that licensing a commercial ISA would lead to problems that would be unsolvable. In addition to the problems discussed in Section 4, the Army concluded that there would be inadequate competition for military computers using the commercial ISA. Potential bidders indicated that they did not want to get involved with a competitor's protected ISA for fear of legal difficulties and the inability to use experience gained to their own commercial advantage. Thus, in February 1980, the Army adopted the use of a government-owned ISA as a primary approach. Public announcement that Nebula had been chosen as the ISA for MCF was made in March 1980.

5.3.2 The Development and Evolution of Nebula

The design of Nebula was begun at Carnegie Mellon University in September 1979. Several analyses involving the use of algorithms that represented the typical military computing environment provided the basis for the design. Programs for these algorithms had been written in various ISA languages and measurements had been made of the efficiency of each ISA in terms of ultimate computer complexity and capacity[10]. Measurements led initially to relative ranking among a mix of seven mainstream government and commercial ISAs.[11] From these tests it was possible to determine desirable and undesirable properties of an ISA with respect to execution of military computations. Later, three more ISAs were evaluated[12]. The design of Nebula, when completed, was also evaluated along with the ISAs of the VAX-11/780 and the CAPS-7. The results indicated that Nebula was a highly efficient ISA[13]. Independent evaluations conducted since the completion of the initial design verified Nebula's high efficiency[14,15].

There was extensive ISA community involvement in the Nebula effort. In December 1979 and January 1980, in-house reviews of the preliminary design were conducted with all three Services represented. A public seminar held in March 1980 was attended by 190 people from industry, government and academia. Comments were solicited. A large number were received, and many of the ideas were incorporated into the design. Ada experts made major contributions. A second industry review was solicited in the Spring of 1980, based on a draft specification. The response contributed to and was integrated into the publication of the first Nebula standard, MIL-STD-1862, dated May 1980.

During the Summer of 1980 discussions on Nebula were held between the Army and the Air Force. The Air Force has been, and plans to continue, using its 16-bit standard ISA, MIL-STD-1750, in avionics systems, but the need for a 32-bit ISA for both ground-based and aerospace systems had also been recognized. The outcome of these discussions was a Memorandum of Agreement that was signed by the Deputy Commanders of Headquarters, Air Force Systems Command (HQ AFSC) and Headquarters, Material Development and Readiness Command (HQ DARCOM) in September 1980, making Nebula a joint Air Force and Army 32-bit standard. The Agreement established a Nebula Control Board, with Air Force and Army membership, to manage the standard. The Agreement assigns the Army responsibility for the development and maintenance of a Nebula Validation Facility, and establishes cooperation on Nebula-related support software. Seven organizations from each Service were placed on the board as voting members. Both services further agreed to evolve MIL-STD-1862 toward the publication of a jointly acceptable standard.

After the Agreement was signed, Nebula was put through many public and government reviews. The Nebula Control Board formed a technical Tiger Team in November 1980 to solicit and consolidate recommendations for change. Many recommendations came from an in-depth review by the Electronics Industries Association which was completed in the Spring of 1981. Others came from ongoing reviews by Ada experts, in-house Air Force and Army personnel, MCF hardware and hardware-support contractors, and from several software contracts. This activity culminated in the publication of a joint Service standard for Nebula, MIL-STD-1862A, which was approved by the Nebula Control Board and issued in September. 1981. Between September 1981 and December 1982 the ISA was fine-tuned and then published in January 1983 as MIL-STD-1862B, the standard that will be employed in all Army and Air Force implementations.

5.3.3 Technical Highlights[16,17,18]

Nebula is a byte-addressed 32-bit ISA. The more important of its architectural features are:

(1) It is a high-level building block approach to instruction structure that simplifies compilation, supports pipelining, supports the generation of compact efficient code, requires much less memory, and facilitates faster operation of programs than most other ISAs. Simple and frequently used instructions have the most compact format and will compute faster than other instructions in the ISA.

(2) It is a high-level procedure control mechanism with matching stack structures that applies not only to procedure calls but uniformly to supervisor calls, interrupts, traps, exceptions, unimplemented operation codes, and illegal operation codes. This feature provides the potential for high execution efficiency and reduction of software errors. Nebula's procedure control mechanism supports implementation in a range of hardware capabilities from micro-computer to mainframe in that full software compatibility of low-end members of the computer family can be achieved without additional cost through use of the OPEX feature. OPEX efficiently initiates software procedures that perform "unimplemented" instructions with exactly the same results, perhaps at slower speeds, than had they been implemented in the hardware of the machine.

(3) It has a virtual memory management system is built into the Nebula instruction set. Efficient relocation and protection of variable-length segments that are user-defined provide the motivation for this system. At the outset, it was required that Nebula's memory management approach be efficient during use in real-time systems where typical memory management systems have to gain needed performance, as well as in command and control systems and intelligence systems in which tasks must be managed and protected in a manner similar to that required in general processing environments. In order to achieve efficient operation, Nebula's addressing and memory management approach contains the following:

(a) Efficient communication between the supervisor and task address space for systems with a single supervisor and multiple tasks.
(b) Protection of the supervisor from improper access by any of the tasks.
(c) Relocation of translation of addresses to allow allocation of contiguous virtual addresses to noncontinguous physical memory.
(d) The ability to divide tasks into several distinct segments for storage and to protect these segments according to function in order to reduce the possibility of internal task error.
(e) Protection of the segments of a task from access and corruption by other tasks in the system. The ISA requires that the hardware automatically provide eight levels of access restriction.
(f) The control of sharing of segments between tasks.
(g) The ability to disable address relocation while maintaining the pro-

tection functions of the memory management system in systems where all tasks reside continually in main memory.

(h) Minimizing the amount of data that must be loaded in order to change tasks. All of Nebula's virtual-address memory management functions are performed by the hardware but no specific implementation approach is prescribed in the standard. For example, in the case of a memory map search, the computer designer may choose hardware approaches such as performing a sequential microcoded search, performing a cache-like search or performing a fully associative comparison that checks all segments concurrently. In any case, the details of the mechanism will remain invisible to the software.

(4) Separate input-output controllers (IOC) with a dedicated IOC instruction set that provides protected virtualized addressing. The IOC instruction set directly supports several types of I/O channels.

(5) A sophisticated exception handling mechanism directly matched to Ada's features in this area.

(6) Direct support for multi-tasking in order to improve efficiency of real-time priority-interrupt-driven systems.

(7) Features optimized for Ada.

(8) Support for multi-level security.

(9) A 32-level priority interrupt structure that supports vectored interrupts.

(10) Four clock-timers included in the ISA.

(11) Support for vectors, arrays, records and more-complex data structures.

(12) Support for the IEEE Floating Point Standard.

5.3.4 Exception Handling

Nebula is also designed to incorporate extensive exception handling facilities--events that cause suspension of normal program execution. Examples are array out of bounds, division by zero, illegal procedure parameter, floating-point overflow, and range error.

Most computer systems simply stop when such exceptions occur and wait for an operator to intervene. In real-time military systems, this approach is unacceptable, and the system must "recover" so that the mission can continue. The philosophy in Nebula, which is a direct reflection of the approach taken in Ada, is that exception conditions, whether caused by the hardware or the software, should be anticipated and the system design should force such conditions to be handled by the software wherever possible.

In Nebula, exceptions can be handled locally within the context of the currently executing procedure; can be raised in the caller's context which is at the level of the procedure that called the currently executing procedure; at higher levels within the task context-stack and in the supervisor's context. A bit in the current program status word will indicate whether exceptions that occur during the execution of the current procedure are to be handled within the task context or by the supervisor. If the former, and if an exception handler is defined within the current procedure, then this exception handler will be initiated and the exception will be handled within the current scope. If an exception handler is not defined within the current procedure, then the exception will be "propagated" (i.e., raised to successive calling levels) until either an exception handler is found or the base (main) procedure (at the bottom of the task context-stack) is reached. If there is no handler in the main procedure, then the supervisor's exception handler shall be called. The transfer to the supervisor's exception handler looks like a procedure call with the exception-code and offending instruction as parameters. This handler may choose to take corrective action, abort the task, return the exception to the offending procedure, or simply continue. Instructions in Nebula permit declaration of an exception handler, raising an exception locally, raising an exception in the encompassing context, determining which exception occured, and storing the address of an exception handler.

5.3.5 Nebula Support/Features for the Ada Language

As stated earlier, a fundamental requirement was established that Nebula efficiently implement as many Ada Features as feasible. Nebula supports all Ada data types (integer, fixed-point and floating-point). It efficiently implements Ada procedures for program structuring using context stacks for user and supervisor tasks. Procedure nesting is accomplished using stacks. Nebula incorporates the use of explicit procedure descriptors and has an efficient parameter passing mechanism built into its CALL and RETURN instructions. Each procedure is automatically given a new set of registers. Exception handlers may be defined and exceptions can be raised and propagated as incorporated in the design of Ada. Nebula's tasking structure supports Ada's multi-tasking features. Tasks may be created, started, stopped, saved and switched in Nebula as permitted in Ada. A full memory management facility is given for each task wherein each task is granted its own memory map. Nebula's PUSH and POP instructions are intended to ease the Ada-Nebula code-generation effort. Code generated from Ada can be efficient since Nebula permits variable-sized instructions and operands and guarantees no side effects of one instruction on another. Further, Nebula's instruction set contains many high-level instructions that match Ada statements. Examples are CASE, LOOP, RANGE, CALL, RETURN, RAISE and START TASK. Overall, Nebula's features reduce the semantic gap between the high order language level and the assembly language levels which offers the potential for reduced software development and post-deployment support costs.

5.3.6 Nebula Support/Features for Military Security

Nebula includes two key characteristics necessary for the implementation of multi-level security: virtual addressing and protection states. The virtual addressing and virtual memory characteristics establish separate address spaces for each task. Supervisor address space is protected from task access. Memory is segmented in the memory management approach and, through segment mapping, each address space is separated and protected. Nebula's protection states include supervisor and user states through which access to executive functions is controlled. In addition, separate kernel and task contexts permit isolation of system-related activities. Privileged executive functions can be separated into a small, verifiable kernel. Transition between protection states can be controlled by a privileged kernel. Input/ Output (I/O) is made secure by adoption of the same characteristics: transfers are to segments in virtual space and the task specifies transfer operations. Input/-Output controllers provide programmable direct memory access transfer. Task execution can be controlled by time, number of instructions and interruptable long-instructions. Privileged instructions are trapped for handling by the supervisor.

5.4 Army Standardization Approach

5.4.1 Background

The computer has become an essential ingredient of almost all Army battlefield systems. Now being employed in increasing numbers, it performs a wide range of functions in the areas of weapon control, command and control, communications, intelligence analysis, navigation, surveillance, target acquisition, sensors, electronic warfare, and combat support services. The computers that are incorporated into manpacks, projectiles, tracked vehicles, aircraft, jeeps, and mobile shelters must be able to function world-wide in extreme environments encountered in deserts, jungles, the Arctic, in the ground and in aircraft.

The rapid growth in the Army's use of embedded computers over the last six years has resulted in extensive proliferation of different and incompatible types. In 1979, a study showed 35 different types of computers employed in a total of 49 Army battlefield automated systems. In 1982, another study showed these numbers were 50 and 65 respectively. Army management became seriously concerned about this situation because of its adverse impact on (1) system survivability, (2) the cost and complexity of hardware logistics support, maintenance, training and acquisition, and (3) the cost of software development and support. An Army survey conducted in the fall of 1981 indicated that there were 131 Army battlefield systems, some deployed, but most undergoing development, in which the computer was an essential, integral element. This points to the potential for even further proliferation of computer types.

The ability of automated systems to perform their missions effectively throughout the length of a battle depends, in part, on the availability of spares and competent maintenance personnel. Further proliferation exacerbates this situation when supply lines are cut or when soldiers and equipment start to become casualties. To assure operational availability of these systems, the Army must carry a large number of unique types of spares, each in relatively small quantities. Also, soldiers are required to maintain the systems in the field. It is too much to expect each such soldier to be able to diagnose and repair fifty different types of computers.

The proliferation of types also greatly increases the cost of logistics and maintenance support of field systems. Factors contributing to this include the development and maintenance of an Integrated Logistics Support (ILS) System for each different computer type employed, reliability testing, inventory management, test equipment, tools, ATE software, maintenance training, operational readiness float, spare plug-in modules and repair parts. ILS development alone typically costs $1 to $5 million per computer type and takes two to three years to complete. Yearly maintenance of the ILS package is approximately $250,000 per different type.

Even though the number of "in-system" fielded computers would be the same with or without ISA or design proliferation, the volume of spares, repair parts, and floats required are increased considerably due to the proliferation of types. There is a minimum level of spares, parts, and floats is required for each unique type.

A preliminary analysis, taking into account the cost of ILS development and maintenance, LRU spares, the maintenance float, spare piece-parts, reliability testing, and maintenance training, predicts that the cost of continued proliferation would be in excess of $700 million per year. This analysis included only part of the potential savings. Additional cost savings can be made in management, test equipment and tools, field generated Engineering Change Proposals (ECPs), software support, and the benefit derived from the use of a small number of common computers and software products.

The Army is convinced that computer proliferation reduction will reduce the number of items required in the supply pipeline, provide simplified logistics and maintenance support, and an order-of-magnitude improvement in damage/casualty backup capability. The Army recognizes, however, that its approach must avoid technology obsolescence and must provide for competition on a long-term basis. These are particularly difficult to achieve in the computer field where techology is advancing so rapidly and where the legal protection of software and ISAs invariably has led to perpetual use of individual vendor's equipment on a basis of less than open competition.

In an effort to address software issues, a task force was created by DARCOM in July 1978 to look into the requirement for Post-Deployment Software Support (PDSS). The results of this study of 91 Army systems indicated 58 different computers, 43 different languages and almost 91 different host support systems were in use or planned. This study was the last of a series and indicates the same result. It was clearly time for the Army to develop a consolidated plan. The PDSS Concept Plan established 11 software support centers in the continental United States. These 11 centers represent a significant reduction of host support facilities that would continue to emerge if such a consolidation were not made. One criterion used in the recommended consolidation was the recognition of battlefield functional areas. PDSS Software Support Centers group together similar systems, such as Fire Support Artillery, at one location so that system application engineering expertise can be better shared and interconnections between systems can be better achieved where required to fit the software.

Preceding studies of the Post-Deployment Software Support, the Army worked with OSD toward the goal of HOL standardization and toward the initiation of the Ada Program. In addition, the Army worked closely with OSD to develop a set of requirements for a programming environment associated with the Ada language.

Another Army software problem has been in the area of embedded computer operating systems. For years, most of the developers of military embedded computer systems have also developed their own operating system. Although these operating systems were highly customized for their applications and usually exhibited high performance, the cost and risk for the development of these operating systems has been very high. Such resource control software has been shown to be very difficult to develop and inherently the most error prone of all software. The development of these unique operating systems is causing the Army to incur tremendous risks, schedule delays, and additional unwarranted costs. This proliferation of operating systems is also causing a deleterious effect on the ability to perform the required post deployment support of Army computer-based systems, due to the uniqueness of these control programs and the fact that they are often embedded in the applications software.

In 1982, a project was initiated for a multi-level secure operating system to complement the Army's Ada and MCF effort. The Army has been closely associated with the OSD-initiated Ada Program since its inception in 1975. The Army has supported the activities necessary to define the language and is now in a direct role supporting the implementation of Ada Language. From an Army introduction point of view, initiatives necessary to transfer this technology to operational use within Army weapons systems and within the electronic industry at large have begun. These initiatives are in support of the Army commitment to begin use of Ada in the 1983-84 time-frame. Army studies have defined the computer resource management issues and solutions to the burgeoning software support costs that the Army is currently experiencing and the role that Ada and the standard programming support environment will play in bringing this problem under control. It was a natural outcome that the Army's commitment to Ada was seen as the major technology answer to the PDSS problem and that it should be adopted along with the resource reallocation of personnel and capital equipment to 11 software support centers. In June 1980 the Army awarded a contract to Softech, Incorporated, Waltham, Massachusetts, to develop the Ada Language System (ALS) of tools.

5.4.2 Computer Acquisition Strategy

The Army's acquisition strategy is structured to provide a solution to the problems caused by excessive proliferation of different types of mission-critical computers while dealing explicitly with the critical issues surrounding advancing computer technology and the need to achieve and maintain effective competition. In the following discussion, these issues will be addressed in an integrated fashion.

The essence of the approach is the development of and budgeting for a long-term acquisition plan that addresses both the need and the critical issues before the commencement of procurement. Analogous to the pre-planned product improvement (P^3I) philosophy, but on a larger scale, the Army has planned multiple, time-phased, competitive acquisitions for common computers for use in those environments where operational availability is important and common spares make sense. This approach contains a philosophy that, on first appearance, seems to be self-contradictory: Standards must change over time, some rather quickly. HOL, ISA, and hardware and software products will have to change and the need for such changes is anticipated at the outset and is the key to ultimate long-term success of the Army's program. Hardware will be changed fastest. Specifications for ISA and HOL will last a good bit longer. Nebula, the initial ISA, will evolve and eventually will be superceded. The same holds true for Ada. In each case, the old will co-exist with new, at least for some appropriate period of time. The hardware cycle will be of the order of five to eight years, the ISA cycle 15 to 20 years, and the HOL cycle 20 to 25 years. As the industry matures into the twenty-first century, these cycles are expected to increase intervals. However, the exact times are unimportant--what is important is the relationship between each standard and the prevailing state of technology surrounding it.

Computer developments will be competitive with short development periods of about five years, the time it usually takes industry to complete total development and preparation for production of a typical commercial product. Once development has been completed, products will be manufactured by multiple producers and fielded. Competition during production will take place either between manufacturers of identical units or between manufacturers of units that are form-fit-function equivalent but that employ different internal designs and technologies. In either case, competition will be limited in order to reap the benefits of commonality discussed earlier. While the field-life of a product may be of the order of 15 years, the primary production period (excluding the production of long-term spares) will be limited to from five to eight years. At the end of the primary production period, a new primary production period will commence, based on the outcome of the competition with possible development.

This approach addresses technology and competition, in that each successive development will involve a fresh round of competition and the opportunity to employ new technologies in the ensuing production models. Advanced technology products will be software and interface compatible with its precedessor generation but are expected to have improved reliability, maintainability, power, size, weight, cost, speed and memory capacity. Future units will maintain instruction set (software) and interface compatibility with units produced in previous generations in order to provide the potential for upgrade/-replacement of older units in the field. Support for older fielded units, however, is expected to continue for as long as it is practical to do so.

The execution of this approach has begun with the first development now underway. The approach to competition and advanced technology being pursued in the first phase will now be discussed. Following industry review of draft specifications and goals for production units, an open solicitation was released that was responded to by 12 bidders representing fifteen organizations. Awards for advanced development were made to GE/TRW, IBM, Raytheon, and RCA in May 1981 with pre-defined areas of competition and evaluation priorities established in order to keep the competition in focus. Areas of competition include technology, hardware architecture, reliability, maintainability, life-cycle costs, power consumption, size, weight, producibility, milspec environment, speed, and memory capacity. The major evaluation priority order is as follows:

(1) reliability and maintainability,
(2) life-cycle cost and power,
(3) size and weight, and
(4) speed and memory capacity.

In March 1982, IBM was eliminated from the competition. Advanced development models have been delivered for testing. After the prototypes and long-term approaches have been evaluated, two of the original four contractors will be selected to continue the competition through full-scale engineering development.

With regard to the technology that will ultimately be employed in the first production phase, there was the danger of obsolescence, even with a relatively short development cycle. If, for example, technology approaches were forced to be pinned down in the bids for advanced development and such technologies became the basis of advanced development models which were then finalized during full-scale engineering development, the probability would be high that production units (five years later) would be close to technological obsolescence. Rather than to evolve the technology in this manner, the Army has taken a different approach.

During advanced development, two technology efforts are being pursued by each contractor. The first effort is oriented toward the use of 1981 technology in prototypes. The second is oriented toward selection of technology for production. The latter requires the development of Technology Insertion Plans that will contain analyses and projections of potentially suitable technologies that will be feasible for use in FSED models in 1984.

Contractors have been given a free hand to choose the approach to technology as well as to hardware system architecture. Technologies currently being pursued are Bulk CMOS, CMOS/SOS and Triple Diffused Biopolar. Each of these technologies are being pursued by companies participating in the VHSIC program. The MCF contractors are closely tied to the VHSIC program. For example, Raytheon has subconstracted to Harris Semiconductor for Bulk CMOS technology; TRW is a VHSIC contractor pursuing triple diffused bipolar technology; and RCA, a leader in CMOS/SOS technology, is linked to Hughes Aircraft, a VHSIC contractor pursuing CMOS/SOS technology. To complete the entire development in five years, the Army has planned short prototype and full-scale development (FSD) phases with short test and evaluation periods at the end of each phase. Integrated Logistics Support (ILS) will be completed during FSD by the two contractors competing that phase.

5.4.3 Army Policy on Standardization

In July 1980, following an in-depth review by an Army Science Panel, the Army staffed a coordinated policy statement which was signed by the Assistant Secretary of the Army (RDA). Now implemented in AR 1000-1, this policy on standardization with respect to battlefield automation systems requires the Army to control the proliferation of computers, HOLs, and instruction sets. For the long-term, it directs the use of a HOL, the use of a single instruction set (now called Nebula), and the use of a family of no more than two military computers. This policy is under review to insure that premature standardization is not levied on the Army.

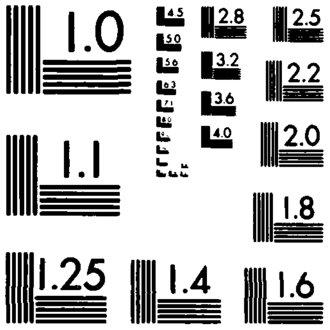
5.4.4 Rationale for the Structure of the Computer Family

In the 1977-78 time frame it was thought that four MCF computers would be required to satisfy the spectrum of requirements for Army battlefield automation systems: a microcomputer, a low-performance minicomputer, a high-performance minicomputer and a maxi-computer. However, in 1979, review of market surveys conducted for the MCF Program by Control Data Corporation led to the conclusion that there was no real need for a maxicomputer for battlefield use. It was also concluded that the micro requirement could be satisfied effectively via a single board computer and components (chip set) from the low-end member of a two-member family. The drive was to reduce, as much as possible, the number of distinct types of logistics items to be supported on the battlefield and still satisfy future requirements effectively without using too large a computer when a smaller one would do the job.

AD-A132 899 DEPARTMENT OF DEFENSE COMPUTER TECHNOLOGY A REPORT TO CONGRESS(U) OFFICE OF THE UNDER SECRETARY OF DEFENSE FOR RESEARCH AND ENGINEERING WASHINGTON DC AUG 83 2/2

UNCLASSIFIED F/G 9/2 NL

END


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

These studies concluded that a computer was needed that could handle high-end applications like TACFIRE, PATRIOT, AN/TSQ-73 Air Defense System, and the TRI-TAC Message/Circuit Switches, (AN/GYK-12 class machine) and another computer at the low end of application where computers such as the AN/UYK-19 were being used. The first became the MCF AN/UYK-41 and the second the MCF AN/-UYK-49. There also appeared to be many applications in which the computer was required to be an integral component of an equipment chassis. A single board computer could satisfy such a requirement. Rather than introduce a third unique logistics item to the MCF family, a single-board computer could effectively be one of the component cards of the AN/UYK-49 (a 6" x 9" card, suitable for use in standard Air Transport Rack (ATR) cases, would be optimum). Both the AN/UYK-41 and AN/UYK-49 were thus specified with this requirement. Subsequently, discussions held with the US Army Missile Command and the US Army Armaments R&D Command led to the requirement for yet another capability-a self-contained micro-computer on a round card for use in missiles and other projectiles. There has not been a resolution as to whether one or two such devices are necessary but, in either case, it is expected the round version(s) will be another form factor.

In many "deeply embedded" applications, only a chip set can be used. For reasons of size, form factor, etc., single board units are inappropriate for such systems. Where there exists a requirement for life-cycle software support, however, use of MCF chip sets, i.e., chips that are component parts of the AN/UYK-41 or the AN/UYK-49 may be cost-effective, especially since commonality at the ISA (Nebula) level would be maintained.

In summary, the analysis lead to the structure of a family containing two primary members, the AN/UYK-41 and AN/UYK-49, and several derived members: the 6" x 9" single board computer, one or two round single board computers, and two chip sets. It should also be pointed out that the added provision for both loosely coupled and tightly coupled distributed processing configurations of the family members provides the ability to finely match to requirements or to expand beyond the maximum capability of the high performance member, the AN/UYK-41. This approach maximizes the logistics commonality and software commonality while at the same time providing a sufficiently wide range of capabilities to capture most of the Army's battlefield requirements.

The last issue addresses the performance and physical requirements for each family member. As discussed above, technology that will be mature in 1984 will be used in MCF. Technology forecasts developed in 1979 by the MCF Project Office and by EG&G, Inc. pointed to the expected feasibility of 20,000 gate VLSI logic chip and 64,000 bit memory chips. EG&G completed a rough design of the AN/UYK-41 and it was concluded that a "super-mini" class machine could be housed in a single ATR size case (7 5/8" x 9 3/8" x 19"). An effective balance was desired among size, reliability, maintainability, power, speed, memory capacity, and cost, all of which are interrelated and could be traded off against one another. The approach taken to converge on these parameters was similar to the Delphi technique. The Army first postulated parameter values and then, through an extensive series of briefings and discussions within DOD and with industry, moving between that which was desired and that which was considered feasible, converged on a set of goals that were incorporated into preliminary specifications. These were formally released to industry for comment, refined where necessary based upon the industry response, and then made a part of the MCF procurement as prioritized goals for the full-scale development models (not for the advanced development models).

Each of the goals represented highly desirable improvements over current capabilities based on anticipated future system needs. Approximate relative improvements that the goals represent are: 5 to 1 improvement in reliability, 4 to 1 reduction in power, 5 to 1 reduction in size, 10 to 1 improvement in memory capacity, 5 to 1 improvement in speed, 5 to 1 improvement in cost per megabyte of main memory. Unit production cost was assumed the same as that for 1980 computers in spite of the increased capabilities. Once the goals for the AN/UYK-41 were established, the goals for the AN/UYK-49 and the single board computer readily fell into place.

Goals for the AN-UYK-41, the AN/UYK-49 and the single board computer are summarized in the following paragraphs.

5.4.5 Computers

5.4.5.1 AN/UYK-41 Super Minicomputer

The most powerful member of the family is the AN/UYK-41 supermini-computer. The following goals have been established for production models of the AN/UYK-41:

Speed	3 Million Instructions per Second without I/O 2 Million Instructions per Second with I/O
Memory	4 Million bytes
I/O Channels	2 Parallel Point-to-Point Interfaces, 2 Dual-redundant MIL-STD-1553B Interfaces, 2 Serial Point-to-Point Interfaces, and 2 Parallel Busses.
Interrupt Response	5 Microseconds
I/O Processor Aggregate Thruput	6 Million bytes per Second
Cost	$75,000 (1980 dollars)
Reliability	10,000 Hours Mean Time Between Failure
Maintainability	30 Min Mean Time to Repair
Volume	0.52 Cubic Feet (short Air Transport Rack (ATR) Case)-7 5/8" x 9 3/8" x 12 9/16"
Built-in-Test	98 Percent Detection, 95 Percent Isolation to LRU, 98 Percent Isolation to 2 LRUs
Power	100 Watts
Weight	40 pounds

5.4.5.2 AN/UYK-49 Microcomputer

The second family member is the AN/UYK-49--a stand-alone micro-computer. The following goals have been established for the production units:

Speed	500,000 Instructions per Second without I/O 100,000 Instructions per Second with I/O
Memory	1 Million Bytes

I/O Channels	1 Parallel Bus 1 MIL-STD-1553B Interface 2 Serial Point-to-Point Interfaces 1 Parallel Point-to-Point Interface
Interrupt Response	40 Microseconds
I/O Processor Aggregate Thruput	2 Million bytes per second
Cost	$25,000 (1980 dollars)
Reliability	33,000 Hours Mean Time Between Failure
Maintainability	15 Min Mean Time To Repair
Volume	0.25 Cubic Feet (1/2 Short ATR Case)- 7 5/8" x 4-1/2" x 12 9/16"
Built-in-Test	98 Percent Detection, 95 Percent Isolation to LRU, 98 Percent Isolation to 2 LRUs
Power	25 Watts
Weight	20 Pounds

5.4.5.3 Single Board Computers

For those systems that do not require the capabilities of the AN/-UYK-49, it will be possible to use one of the two standard MCF single board computers. The first single board-computer will be a 6" x 9" micro that will be one of the boards (LRU) from the AN/UYK-49. The second singleboard computer will be a missile/projectile micro on a round card that will be built using chips from the AN/UYK-49.

First-phase goals for production models of the 6" x 9" single board computer are:

Speed	500,000 Instruction per Second
Memory	128,000 Bytes
I/O Channels	1 MIL-STD-1553B Interface and 1 Serial Point-to-Point Interface and System Interface
Cost	$5,000 (1980 dollars)
Reliability	100,000 Hours Mean Time Between Failures
Volume	0.02 Cubic Feet (1/2" x 6" x 9")
Power	5 Watts
Weight	12 Ounces

Requirements for the round single board computer in the area of memory, I/O channels, and size will be established in advance of initiation of fullscale development.

5.3.5.4 Chip Sets

For those battlefield system requirements where either of the two standard family members of either of the single board computers would tend to be overkill, MCF software compatibility could be preserved using VLSI chips that are components of the AN/UYK-41 or the AN/UYK-49. Such chip sets may be applicable to systems for which the form factors or capacities of the standard units described above are not suitable. Systems able to use these chip sets in specialized (non-MCF) LRU's would be able to benefit from the standard software support and from MCF production runs.

5.4.5.5 Computer Control Panel

The functions performed by the Computer Control Panel (CCP) with respect to the AN/UYK-41 and AN/UYK-49 are control, diagnostics, and software development support. The CCP also performs its own selfdiagnostics. The CCP contains the AN/UYK-49 Single Board Computer, a mass memory device, a display and control section, an interface section and a power supply. Characteristics and goals for the CCP are:

Interfaces	Maintenance Interface Serial Point-to-Point Parallel Point-to-Point Serial Bus (MIL-STD-1553B)
Mass Memory Unit	Removable Media At Least 2 Million Bytes Storage
Weight	Less than 50 pounds
Power	Less than 100 Watts
Reliability	10,000 hours Mean Time Between Failures
Maintainability	30 minutes Mean Time to Repair

When connected to the AN/UYK-41 or AN/UYK-49, the CCP will be capable of switching the computer from its normal operational mode to bring it under control of the CCP. The CCP will provide the ability to modify or continuously display any portion of the operational state of the computer to which it is attached. "Operational State" includes all user and non-user accessible registers, the contents of memory and the run/halt state. The CCP will provide the capability to selectively record the operational state in its mass memory. The Computer Control Panel, in conjunction with the AN/-UYK-41 or the AN/UYK-49 Built-in-Test (BIT), will be capable of correctly isolating all single hard failures to one Line Replaceable Unit (LRU) for 95 percent of the failures and to no more than two LRUs for 98 percent of the failures. The Computer Control Panel will be designed and implemented to lead personnel step by step through AN/UYK-41 or AN/UYK-49 diagnostic procedures. This will be accomplished by a series of clearly and legibly displayed instructions and prompts. The fault dictionary and diagnostics will be contained on mass memory media. The internal logic of the Computer Control Panal will address the fault dictionary and perform the fault symptom/-LRU cross-referencing. Only a replacement statement concerning the target (LRUs) will be displayed. In the event that more than one LRU is called out in the replacement statement, the Computer Control Panel will indicate the most likely candicate for replacement.

5.4.6 Survivability

There has been a strong emphasis on computer and system survivability in the MCF Program due to the degree of future dependence of fighting forces on automation. The need for survivability has provided one major impetus for fielding a family of standard, software-compatible computers. Should a high-priority system malfunction during a battle because of a computer failure, if parts or replacement computers are not available due to a shortage or due to cut lines of supply, then it would be possible to restore the operation quickly by taking parts or entire computers from lower priority systems. Further, standardization will facilitate the completion of repairs quickly due to the use of common parts and the availability of maintenance personnel.

5.4.7 Summary

The current Army programs to reduce proliferation have been based on numerous studies and analyses over the last eight to ten years. The Ada Language System (ALS) development aims at reducing support software and host system proliferation in the PDSS centers and the introduction of Ada in concert with DoD and the other services. The MCF program has been designed to pursue competition and technology insertion, while reducing proliferation of hardware to reduce maintenance and training costs. Until these products are available and tested, the Army will control proliferation by using existing support systems and hardware products to the maximum extent possible. As the ALS and MCF products become available, common run-time support systems by functional area are planned to reduce proliferation of system software to the extent feasible.

5.5 The Air Force Approach

5.5.1 Overview

Planned Air Force embedded computer instruction set architecture (ISA) standardization efforts will build upon a logical extension of on-going current policy involving MIL-STD-1750A 16-Bit and MIL-STD-1862A 32-Bit ISAs. This approach fosters competition and encourages technology insertion by treating the ISA as an interface between programmer and machines, or between compiler or assembler and machine. By standardizing the ISA, the Air Force can exploit a common base of software support resources: people, facilities, and tools. Commonality at the ISA level permits competition among vendors during development, in production, and in subsequent system modifications. This is accomplished by leaving the responsibility for the computer hardware specification (i.e., size, weight, cooling, connector type, shape, etc....) with the individual system program office and/or the prime system contractor while leaving the responsibility for design, manufacture, processing techniques, and device technology with the computer vendors. This approach allows for transfer or reuse of software to the maximum degree economically feasible while fully realizing the benefits of competitive pricing and latest state-of-the art proven technologies in acquiring avionics computers. This has already yielded improved designs, technologies, and more competitive pricing in several system acquisitions, including the B-1B, the F-111 Digital Upgrade, the F-20 fighter aircraft, and the Wild Weasel upgrade programs.

Although the Air Force logistics environment differs markedly form the other two services, additional levels of standardization beyond the ISA may prove valuable. Particular emphasis will be placed on standard interfaces, such as MIL-STD-1553 and MIL-STD-1760. Typically, hardware standards are most effective in areas experiencing slow technological growth. Continued search for opportunities for form-fit-function (F^3) or hardware standards is part of the total Air Force computer resources standardization program, which is not limited solely to embedded computers.

5.5.2 Policy

It is Air Force policy to standardize on the MIL-STD-1750A ISA for all processors in Air Force defense systems when technically practical and cost effective over the system life cycle. It is the intent of the Air Force that MIL-STD-1862 (when available) be considered for all embedded systems where long word (32-bit) militarized computers are required. This policy allows for the following exceptions:

a. Commercial computers used in support of program development or administrative, and procured in accordance with either Air Force 300-series or 800-series regulations.
b. Non-militarized, general-purpose, commercially available automatic data processing assets.
c. Digital computers and processors used in systems for which there is no anticipated need to maintain or modify software over the system life-cycle.
d. Digital computers and processors used as part of automatic test equipment and aircrew training devices.
e. Experimental use of unconventional or advanced technology processors in basic research and exploratory development.
f. Processors embedded in standard equipment (as defined in AFLC/AFSC Regulation 800-31).

Processors used in airborne avionics systems and air-launched strategic/tactical missiles must conform to MIL-STD-1750A ISA requirements. Use of commercially available MIL-M-38510 microprocessors not conforming to MIL-STD1750A is authorized until MIL-STD-1750A processors qualified to the required level of MIL-M-38510 are available.

Developers of other Air Force defense systems must consider the use of MIL-STD-1750A ISA for all processors in the system. Consideration must be given to development costs and schedule, performance, and life-cycle costs.

Developers of systems who need a waiver to the mandatory MIL-STD-1750A requirement must base the request on technical, and/or system life-cycle cost, schedule or risk factors. If the waiver request is for the use of a commercial ISA in the DoD inventory, or other DoD-approved ISA, field activity organizations can grant the request. In other cases where a waiver to MIL-STD-D-1750A is requested, the headquarters organization must make the final determination.

5.5.3 MIL-STD-1750A

5.5.3.1 Background

MIL-STD-1750A, "16-Bit Computer Instruction Set Architecture", is the Air Force developed and owned ISA. Developed through work at the Air Force Avionics Laboratory, the ISA was orginally intended for use as the central computer in an avionics suite. Refinement in open forum (industry formed users group) expanded the ISA's memory addressability and improved its ability to effectively execute programs written in modern High Order Languages.

MIL-STD-1750 evolved from the Avionics Laboratory's Digital Avionics Information System (DAIS) program. In 1976, a team of Avionics Laboratory engineers and avionics engineers developed an Instruction Set Architecture (ISA) for use in the DAIS program. This ISA was submitted to industry for their comments thereby starting an active dialogue with approximately 20 companies. When OUSDR&E, concerned with escalating software costs, held their workshop at Fort Belvoir, Virginia, in October 1977, the Air Force presented the DAIS architecture ISA as an initial baseline and then subsequently evolved it into a standard that provided vendor independence and technology transparency. Recognizing the importance of developing a standard acceptable to the industry, the first open forum meeting was held in January 1978. Following two subsequent reviews, the basic MIL-STD-1750 was published on 20 February 1979.

The new standard was placed on two airborne computer development contracts (awarded competitively) by the Avionics Laboratory to demonstrate vendor independence, implementation technologies, and effects on operational as well as support software. Prior to this, in August 1979, the industry organized a MIL-STD-1750 Users Group with approximately forty companies actively participating. Between August 1979 and February 1980, four open forum meetings were held by the the MIL-STD-1750 Users Group. By then, the Air Force had already institutionalized a control structure for the standard to be responsive to the Users Group's needs. The original MIL-STD-1750 was updated (incorporating 47 changes approved by the Users Group) to create the MIL-STD-1750A version of the standard with a release date of 2 July 1980.

5.5.3.2 Competition & Technology Insertion

As soon as the Avionics Laboratories demonstrated the ISA concept in March 1980, many vendors started developing MIL-STD-1750A computers under their internal IR&D efforts. After "priming the pump", the Avionics Laboratory effort was terminated and vigorous private sector competition surfaced. Currently there are at least 20 vendors within the United States and foreign countries who have implemented MIL-STD-1750A using the latest state-of-the-art technologies. Several companies within the United Kingdom are implementing MIL-STD-1750A for their Ministry of Defense (MoD). Israel undertook an ambitious effort to develop MIL-STD-1750A ISA implementation capability in 1982 and is now requesting the Air Force to provide the ISA compliance verification tool for conduction of the first verification in May 1983. MIL-STD-1750A is presently under consideration by the NATO community and a STANAG is being drafted. ASCC Air Standard Working Group Party 50/108 is also considering the ratification of MIL-STD-1750A.

Since establishing the MIL-STD-1750A ISA policy and institutionalizing the control structure, the Air Force has experienced more competition in technology growth and price reduction. Recent experience on several procurements for Air Force systems illustrate that requiring MIL-STD-1750A is forcing the vendors to compete with their best hardware (i.e., latest device technologies, innovative implementation schemes) thereby providing greater performance capabilities, higher reliability, and competitive pricing. It has also forced prime contractors to compete for the best computer buys rather than remaining with their favorite suppliers even though, at best, the products were four to eight years old in technology and reflected higher prices accordingly. The F-111 computer replacement program, B-1B computers, F-20 computers, HH-60

Helicopter program and the Wild Weasel program are a few examples that provide positive testimony for competitive pricing and technology insertion directly stimulated by standardization at the ISA level. It is worth noting that the Wild Weasel program utilizes multi-processing techniques employing a three Central Processor Unit (CPU) configuration to provide three million instructions per second throughput. This illustrates the power of ISA standardization and that it helps rather than hinders system capabilities.

MIL-STD-1750A has been found to be well suited to a variety of Air Force applications. Applications range from the Army Division Air Defense System to the F-16 Multinational Staged Improvement Program. The technologies range from bi-polar integrated circuitry, to VHSIC chips, to radiation hardened CMOS/SOS. However, major technological breakthroughs such as high performance VLSI and VHSIC implementation still require "seed monies" in early technological demonstration stages. The Air Force has funded two such efforts. These are the F-16 VLSI MIL-STD-1750A program and the MIL-STD-1750A VSHIC program.

<u>MIL-STD-1750A in the VLSI (F-16 Program)</u> - In early 1981, the F-16 System Program Office (SPO) awarded a contract for the F-16 Multinational Staged Improvement Program to General Dynamics Corporation. The program involved major updates to the avionics systems/subsystems including the computational subsystems and associated computer hardware and software. Besides MIL-STD-1750A compliant computers for the Fire Control System, this program also involved several microcomputer applications. These applications consisted of Advanced Central Interface Unit (24K 16-bit words), Multifunctional Display Unit (19K words), Data Entry Cockpit Interface Set (10K words), and the Data Transfer Unit (10K words). In addition, two other Air Force programs (i.e., LANTRIN and Heads Up Display under development at the time) targeted for F-16 use were also using MIL-STD-1750A computers. It was only prudent to establish an approach for the F-16 that exploited a common base support resource; personnel, facilities, and tools with specific emphasis on programming support tools such as complier, assembler, linker/loader, and Instruction Set Simulator.

General Dynamics, in conjunction with the Aeronautical Systems Division, conducted a survey of companies building the MIL-STD-1750A compliant computers at the time to find a suitable microcomputer. None of the existing microcomputers within the constraints of size, speed, power, and weight were found to satisfy the system requirements established by General Dynamics. Many other avionics programs and armament missiles were also facing similar dilemmas. The Air Force saw a potential market for MIL-STD-1750A microcomputers for their applications especially when the armament arena was considered.

Thus, anticipating significant long-range benefits to be derived from the adoption of MIL-STD-1750A, the Air Force decided to fund the development of a high performance MIL-STD-1750A microprocessor. With F-16 as the first candidate to use the MIL-STD-1750A microprocessors, the Air Force further decided to fund the development contract through General Dynamics to reduce programmatic risks to the maximum extent possible.

Fairchild Camera and Instrumentation Division won the contract in fierce competition and the vehicle was awarded toward the end of 1981. Fairchild is using Isoplanar Integrated Injection Logic (I^3L)-II Bipolar VLSI technology. This technology affords static operation with 200 nanosecond bus cycle time with inherent radiation tolerance (1 x 10^5 rads), and operation at 20-MHz over the entire military range. The Fairchild microprocessor is a single chip, designated F9450, that completely implements MIL-STD-1750A Instruction Set Architecture requirements. The 64-pin F9450 has sixteen registers to implement interrupts, single-and double-precision arithmetic, as well as 32- and 48-bit floating point instructions on-chip. Real-time processing is achieved through advanced design and architecture, incorporating two programmable timers, a complete 16-level interrupt processor, and a comprehensive fault handler on a chip. Multiprocessing is supported by a flexible bus arbitration technique scheme along with process synchronization (test and set) instructions. Several support circuits provide additional capability. These include F9446 Dynamic Memory Controller, the F9451 Memory Management Unit, providing memory-mapped expansion to one million words; and the F9452 Block Protect Unit. A multi-user development system (FS-1) has been developed along with real-time system emulation and debugging capability.

The die shrink is finished and the prototype parts delivery is expected in the second quarter of 1983 with preproduction quality 100 chip sets due in the last quarter of 1983. The production units are expected to be available in third quarter of 1984. This high performance processor will provide over 700 thousand instructions per second (i.e., based on Digital Avionics Information System instruction mix) which far exceeds its existing military qualified counterpart VLSI implementations. Besides the DoD market, Fairchild also expects a commercial market for this chip. At 0.5 million instructions per second throughput without floating point, the F9450 is expected to compete in the robotics market.

MIL-STD-1750A in the VHSIC Program - From the beginning of the program, VHSIC has incorporated MIL-STD-1750A general purpose computers. Specifically, two of the six Phase 1 (first VHSIC chip generation) contractors incorporated MIL-STD-1750As in their brassboard demonstrators; these are the Texas Instruments Multimode Fire/Forget Missile Guidance Processor and the Westinghouse Multimode Tactical Radar Signal Processor. In both cases, these are truly "embedded" computers in the sense that the chips used to implement the MIL-STD-1750A ISA are laid out as convenient among the breadboard components rather than being segregated as distinct units. Both designs illustrate the power of VHSIC as an implementing technology for ISA concepts because they require only a few chips (on the order of 2 dozen, depending on the memory used) and were readily accommodated with the respective manufactures' VHSIC chip design schemes and overall Phase 1 chip inventories. Moreover, these MIL-STD-1750A assemblies are projected to be several times higher in throughput than physically larger earlier models built with pre-VHSIC components (two tp five million operations/sec vs fewer than one million operations/sec.)

By minor design changes, these and similar embedded MIL-STD-1750A processors can be packaged as stand-alone units. Essentially the same chip sets, supplemented as appropriate with memory one input/output components, can be used. Preliminary design results show that these VHSIC computers will fit on just one or two circuit boards, compared to a full box of circuit boards for earlier technology versions. The widespread acceptance of MIL-STD-1750A means that contractors' design investments will have an extremely high return in operational utility for multiple systems.

VHSIC will also demonstrate the key software development concepts associated with MIL-STD-1862. Both Texas Instruments and Westinghouse are programming their brassboards in structured higher order language (Pascal and Ada, respectively). The use of the general purpose MIL-STD-1750A ISA as an executive processor controlling the function of more specialized digital signal processing elements in these brassboards illustrated the architectural power of segregating the software-intensive control functions from the higher-throughput but algorithmically less complex signal processing functions. The use of the MIL-STD-1750A ISA will greatly simplify the adaptation of these generic brassboards to satisfy the unique needs of particular real-word applications.

5.5.4 MIL-STD-1862A

Because no one Instruction Set Architecture (ISA) can cost-effectively cover the entire spectrum of computer requirements for modern systems, the Air Force has joined the Army as equal members of the MIL-STD-1862A Control Board. The Air Force plans to require either the MIL-STD-1750A or MIL-STD-1862A-ISA for embedded computers. As with MIL-STD-1750A implementation, Air Force logistic requirements will not necessitate the adoption of a servicewide computer hardware standard. Open competition and technology insertion will be encouraged at the individual system acquisition level. However, for applications where the timing of the Military Computer Family (MCF) hardware program matches, it is likely that the Air Force may be able to benefit from quantity buys by the Army.

5.5.5 Miscellaneous Items

The MIL-STD-1750A Users Group--a broad base of government and contractor developers and users--has been instrumental in the user acceptance and improvement of the standard. It is now concentrating on the support resources needed to fully exploit the benefits of the standard. At the most recent meeting of the Users Group, over 120 participants, represented more than 40 different corporations.

The MIL-STD-1750A Control Board, an Air Force body, evaluates all recommendations of the Users Group, and is responsible for control of the standard. The Control Board has committed itself to a three-year freeze of the ISA, as a minimum.

The Embedded Computer Standardization Program Office (ECSPO) at the Deputy for Avionics Control is a joint AFSC/AFLC unit that manages Air Force work to support standard systems with tools. The ECSPO is working to develop and maintain a standard core set of compilers, code generators and other assorted programming tools.

Verification of computers that meet MIL-STD-1750A is underway at the Systems Engineering Avionics Facility (SEAFAC); already 11 firms and 15 computers have passed the MIL-STD-1750A Architecture Test Procedure, with others in line for their turn.

5.6 The Navy Approach

5.6.1 Background

In the 1950s, most of the computers used aboard Navy ships were special-purpose, analog computers designed to solve specific information flow problems. In 1958, the Bureau of Ships designed a digital computer to handle and manipulate the operational data that had previously been disseminated and displayed manually. This was the start of the Naval Tactical Data System (NTDS). These computers were complete units as they stood and could be configured in variable quantities to meet the differing mission profiles of several classes of ships. This concept has permitted the surface Navy to develop a few general-purpose embedded computers for a variety of systems. Most of the initial NTDS computers (CP-642) are still in the Navy inventory today.

From this modest beginning, the spread of the digital computer aboard ship proceeded rapidly, first with the expansion of NTDS and then into many other shipboard systems. By 1970, it was recognized that a proliferation of different types of digital computers in the Navy would have to be controlled in the interest of efficiency in logistics, training, reliability and maintainability, configuration control, system interoperability, and software support. As an initial effort, the AN/UYK-7 computer, the CMS-2 High Order Language (HOL), and certain NTDS operator consoles were designated as standards. This was followed by the competitive acquisition of a standard minicomputer (AN/-UYK-20), cartridge magnetic tape unit (AN/USH-26), alphanumeric display (AN/-USQ-69), tactical display set (AN/UYQ-21), and the development and control of standard HOL compilers and other support software. Standards were also developed and promulgated for embedded computer systems software documentation and quality assurance in SECNAVINST 3560.1 and MIL-STD-1679. In the meantime, airborne systems proceeded under the "total system concept", i.e., placing the engineering of the entire aircraft, including selection of the embedded computers, in the hands of each prime contractor; however, this fostered proliferation of different types of airborne computers. Development of a standard airborne computer was initiated in 1976, when a contract was awarded competitively for the AN/AYK-14 standard computer, to be used initially with the F/A-18 and LAMPS programs and later with other systems. The AN/AYK-14 specification required capture of the AN/UYK-20 instruction set (by emulation), thereby allowing use of a large portion of the AN/UYK-20 support software.

5.6.2 AN/UYK-44 and AN/UYK-43 Programs

The basic need for tactical embedded computers is to acquire, process, and display in a meaningful manner the vast amount of information required for the effective completion of the mission, and to directly exercise control of sensors and weapons.

Ships, aircraft, and weapons of both the U.S. and potential enemies show clear trends toward multiple threat situations and reduced reaction time. Decision-making aids and weapon control systems must keep pace with the capabilities of the ships, aircraft, and weapons. This generates the need for embedded computer capabilities which serve both the current and future information handling requirements of existing combat systems, provide growth potential for the requirements of new systems, and remain operationally effective and economically supportable over a protracted life cycle.

The four fundamental reasons for developing or converting to a new computer family are to:

1. Accommodate expanded operational requirements
2. Improve reliability and maintainability
3. Achieve lower life-cycle costs
4. Avoid logistic obsolescence

Past Navy standardization efforts have prescribed the use of the AN/UYK-7 and AN/UYK-20 computers for use in shipboard embedded computer systems. More recently, the AN/AYK-14 was developed as a standard airborne computer. The AN/AYK-14 executes a superset of the AN/UYK-20 instruction set architecture (ISA) and uses most of the AN/UYK-20 support software. The AN/UYK-43 Navy-Embedded Computer System (NECS) and AN/UYK-44 Militarized Reconfigurable Processor (MRP) and Computer (MRC) are intended to extend this philosophy further in that the AN/UYK-43 will implement a superset of the AN/UYK-7 instruction set architecture and the AN/UYK-44 will implement a superset of the AN/UYK-20 and AN/AYK-14 instruction set architectures. This approach permits the use of the extensive body of AN/UYK-7, AN/UYK-20, and AN/AYK-14 support software and provides a means to affect the transition of existing systems to a computer of significantly increased performance without necessitating a major rewrite of application programs.

As stated in OPNAV Notice 5200 of 15 December 1981, the AN/UYK-43 and AN/-UYK-44 will be used for all new system developments to take advantage of anticipated lower unit cost, decreased power requirements, decreased size and weight, and increased performance. Also, AN/UYK-7 and AN/UYK-20 computers in existing systems will be replaced when system requirements expand beyond the capabilities of the existing computers and they must be replaced, or where replacement is the most cost effective way of meeting long term requirements. In each case, however, due consideration must be given to all costs associated with computer backfit (including installation, documentation, logistics changes, and training) which may severely impact an extensive backfit decision.

To achieve minimum system life-cycle cost and maximum competition during procurement, the level at which standardization is applied is pivotal to the acceptance of the standard and its success in reducing cost. Examining successful commercial computer vendor product lines shows that families of upwardly compatible instruction set architectures minimize software support cost, while allowing evolutionary implementation of extensions to the baseline in order to incorporate needed capabilities as technology allows or applications demand.

Figure 5-1 illustrates the evolution of the Navy standard 32-bit instruction set architecture, based on that implemented in the AN/UYK-7, and the Navy standard 16-bit instruction set architecture, based on that implemented by the AN/UYK-20 and AN/AYK-14. Figures 5-2 and 5-3 show the relative performance of the current and planned implementations of the Navy standard instruction set architectures.

FIGURE 5-1: Navy Standard Instruction Set Architectures

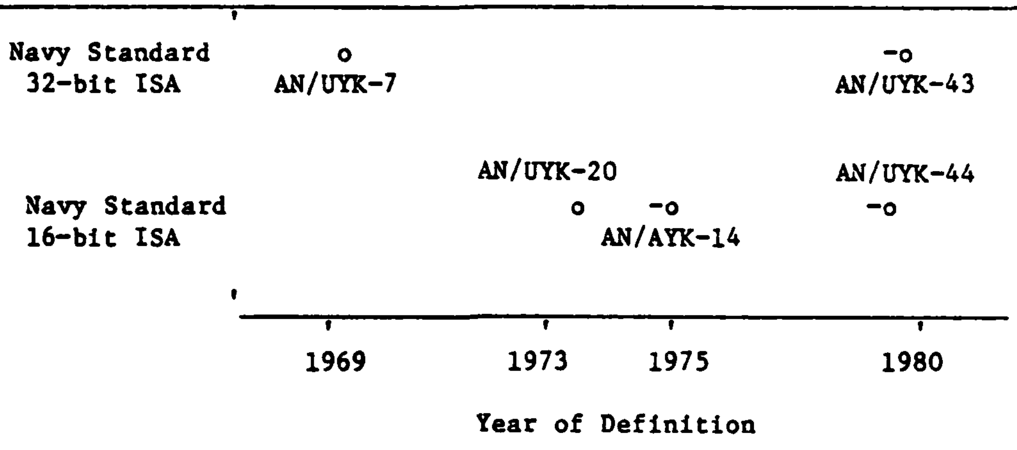


FIGURE 5-2: Implementations of the Navy Standard 16-bit Instruction Set Architecture

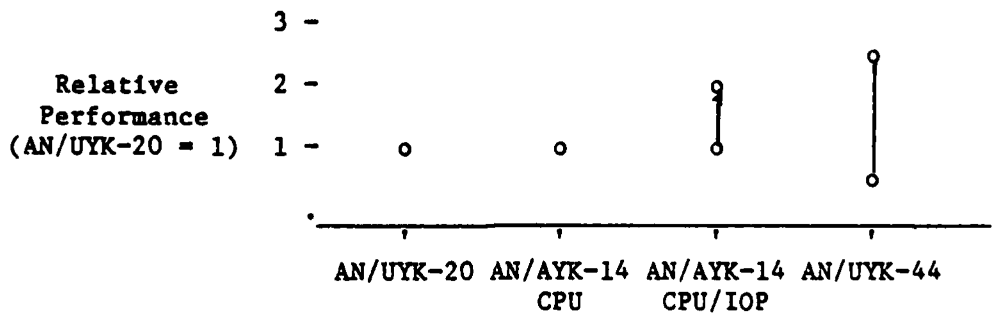
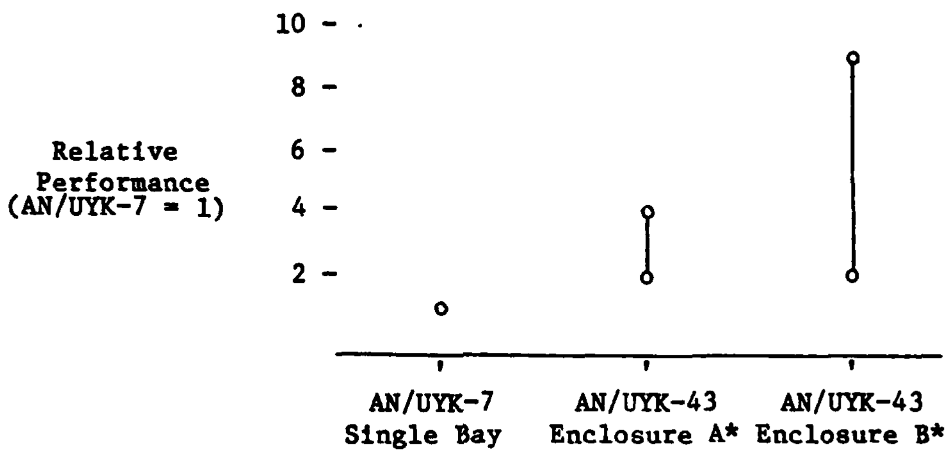


FIGURE 5-3: Implementations of the Navy Standard 32-bit Instruction Set Architecture



* Amplification follows

5.6.2.1 <u>AN/UYK-43 Navy Embedded Computer System (NECS)</u>

The AN/UYK-43 Navy Embedded Computer System (NECS) will be the next generation standard Navy embedded 32-bit computer, superseding the AN/UYK-7 as the embedded computer in Navy digital systems with high performance requirements. The AN/UYK-43 will be a hardware-related family of computers implementing a superset of the AN/UYK-7 instruction set architecture with varying performance and cost ranges. The AN/UYK-43 will have the following attributes:

a. Use of recent substantial improvements in state-of-the-art technology.
b. Design for future technology infusion.
c. Intelligent Input/Output Controller.
d. Performance monitoring capabilities through external receptacle.
e. Incorporation of the AN/UYK-7 instruction set with appropriate extensions.
f. High modularity with attendant ease of modification.
g. Significant improvement of both reliability and maintainability.
h. Extensive debugging aids.
i. Stack register implementation.
j. Computer Interconnection System for connecting up to 16 AN/UYK-43 enclosures.
k. Automatic fault isolation.
l. Capability to be extended to emulate additional instruction set architectures.
m. Both air-and water-cooled configurations.

The AN/UYK-43 with non-volatile memory will have a performance at least 1.5 times that of the AN/UYK-7, and with volatile memory at least 3 times that of the AN/UYK-7. With the high performance option (cache memory) installed, the AN/UYK-43 will have a performance capability of at least 4.5 times that of the AN/UYK-7. In addition, the AN/UYK-43 will have addressability to 4 billion 32-bit words. Reliability is specified as a system MTBF of 6000 hours, and maintainability is specified as an MTTR of 0.25 hours.

The AN/UYK-43 will be available with two different enclosures, each of which will be capable of passing through a 25 inch diameter submarine hatch. Enclosure/A will have the same footprint as the existing AN/UYK-7 and can contain one CPU, one IOC, 24 I/O channels, and 5 memory modules. Enclosure/B can contain two CPUs, two IOCs, 64 I/O channels and 10 memory modules. Each memory module may be 32K non-volatile, 64K volatile, or 256K volatile 32-bit words. The AN/UYK-43 enclosures will be available with air-cooled and water-cooled ordering options and will be environmentally qualified to MIL-E-16400 with a maintenance philosophy consistent with the shipboard environment.

The acquisition strategy for the AN/UYK-43 provided competition throughout the development phase and into the production phase of the project. The benefits derived from this competition resulted in technically superior implementations, with heavy emphasis on reliability and reduced cost for both development and production. Two cost plus incentive fee development contracts for the AN/UYK-43 were competitively awarded to Sperry-Univac Defense Systems Division and IBM Federal Systems Division in September 1980. Competition during the development phase was realized through these parallel development contracts. Competition for the production phase was realized through technical excellence and unit production price competition between the two development contractors.

The following Table presents the AN/UYK-43 development schedule.

TABLE 5-1: AN/UYK-43 Development Schedule

MILESTONE	DATE	MONTHS AFTER CONTRACT AWARD
Contract Award	9/80	0
EDM Delivery	3/83	30
Production Award	5/83	32
First Production Delivery under ALP (PASU)	12/84	51
First Production Delivery under AFP (ASU)	6/86	69

5.6.2.2 AN/UYK-44 Militarized Reconfigurable Processor (MRP) and Computer (MRC)

System design approaches for modern complex multifunction weapon systems, communication systems, command and control systems, and intelligence systems have emphasized the need to process data and signals in a functionally distributed architecture (i.e., processing local to sensors, controllers, data links, launchers, etc.). Functionally distributed systems increase system design effectiveness by reducing the complexity of central processing designs, minimize bandwidth of interfaces by reducing raw data communication requirements, and simplify failure location by automatically correlating failures to losses of specific functions. However, it is unrealistic to envision a system made up of 40 to 60 minicomputers such as the AN/UYK-20 because of cost and space requirements. Therefore, a new and more suitable embedded computer is required. This new embedded computer is the AN/UYK-44 Militarized Reconfigurable Processor (MRP).

The AN/UYK-44 program focuses on the lower portion of the embedded computer family processing requirements. Its market will include applications such as sensors, launchers, smart terminals, controllers, displays, and communication interface units. The AN/UYK-44 consists of an unbundled card set (MRP) and packaged, militarized computer (MRC), which will supersede the AN/-UYK-20. The AN/UYK-44 uses the instruction set architecture implemented by the AN/UYK-20 and AN/AYK-14 with appropriate upwardly compatible extensions to support distributed system implementations and memory management. The AN/-UYK-44 has the following attributes:

a. Use of recent substantial improvements in state-of-the-art microelectronic technology, integrated circuits, and memories.
b. Design for future technology infusion.
c. Use of existing programming tools, MTASS (e.g., CMS-2M crosscompiler, SPL/1 cross-compiler, macro cross-assembler, AN/UYK-20 and AN/AYK-14 simulators, etc.).
d. Functional capability adaptable to system requirements.
e. Implementation with existing components and processes with minimal risk.
f. Form factor adaptable to system requirements.
g. Incorporation of the AN/UYK-20 and AN/AYK-14 instruction set with appropriate extensions.
h. High modularity with attendant ease of modification and a variety of configurations.
i. Mitigation of deviations from standards based on size, power, capability, or environment requirements because of significant flexibility and modularity of implementation architecture.
j. Significant improvement of both reliability and maintainability.

Initial MRP Advance Production Equipments (APEs) were delivered ahead of schedule, beginning in December 1981.

The AN/UYK-44 will be available as the Militarized Reconfigurable Processor (MRP) and the Militarized Reconfigurable Computer (MRC). The MRP will have distinct low performance and high performance options. The MRC will consist of a Basic MRC Cabinet and an Expansion MRC Cabinet.

The acquisition strategy for the AN/UYK-44 provided competition throughout the development phase and into the production phase of the project. The benefits derived from this competition resulted in technically superior implementations, with heavy emphasis on reliability and reduced cost for both development and production. Two development contracts for the AN/UYK-44 were competitively awarded to IBM Federal Systems Division and Sperry-Univac Defense Systems Division in September 1980 based on technically acceptable proposals, lowest prices for best effort fixed price development, and unit design-to-cost goals for production. Competition during the development phase was realized through these parallel development contracts.

Competition for the production phase was realized through unit price, life-cycle cost, technical and qualification factor competition between the two development contractors. Production authorization for a five-year requirements contract was awarded to Sperry Corporation based on significantly lower unit price and life-cycle cost, and successful completion of MRP qualification. Firm fixed prices for MRP, MRC, MRC Basic Cabinet, and MRC Expansion Cabinet for a five-year period have been achieved. Table 5-2 presents the AN/UYK-44 development schedule.

5.6.2.3 AN/AYK-14 Navy Standard Airborne Computer Program

The AN/AYK-14 Navy Standard Airborne Computer is a modularly designed, general-purpose, airborne digital computer which is currently meeting the processing needs for a wide spectrum of applications among each of the services, not only in aircraft, but shipboard and shore bases as well. The overall program has made three major thrusts: basic production, pre-planned product improvements (P^3I), and development of a second source manufacturer. These are all interrelated to form a strong, viable standardization program which can effectively reach into the 1990's with an up-to-date, supportable, affordable general-purpose digital computer.

TABLE 5-2: AN/UYK-44 Development Schedule

MILESTONE	DATE	MONTHS AFTER CONTRACT AWARD
Contracts Awarded	9/80	0
First MRP APEs Delivered	12/81	15
MRP Qualification Completed	12/82	27
MRP Production Authorization	4/83	31
MRP First Production Delivery	10/83	37
First MRC APE Delivered	9/82	24
MRC Qualification Completed	10/83	37
MRC Production Authorization	11/83	38
MRC First Production Delivery	9/84	48

The basic production program has delivered over 700 units to date and is planned for 10,000 units through 1991 at a cost of over $1 Billion. Current users include F/A-18, AV-8B, LAMPS, EA-6B, F-14, E-2C, EP-3, P-3C, ALWT, ALCS, and MIFASS.

The P^3I program has been dynamic and responsive to user needs. To date, memory capacity has been doubled and processing capability increased with no impacts to configuration, software or logistics. Improvements are currently targeted for near term (83-87) and midterm (87-91) projected user requirements while the next generation follow-on computer is undergoing its development and qualification. In the near term, the processing performance can be doubled and made to run efficiently with Ada as the standard HOL. In the midterm, technology insertion of VHSIC can gain across the board improvements in performance (3X) and reliability (8X). These two steps can be made transparent to the users and thus presents a significant savings in their software, their logistics pipeline and in their backfit capabilities for upgrading with minimum program impact. For these reasons the AN/AYK-14 has been identified as a prime candidate for the current DoD VHSIC insertion program.

The second source program has been planned from the outset to provide cost savings from competition and to provide the benefits of a mobilization base in case of national emergency. Current approved plans have initial production competition beginning in FY86 with an overall cost savings of $70M through FY91. The second source producer will be included in the P^3I strategy and is expected to provide competition for each of the improvements.

To allow the AN/AYK-14 to be a viable standard for the latter half of the 1980s, the Naval Air Systems Command (NAVAIR) is now in the process of using a P^3I to update the AN/AYK-14. Probable new modules include a single card processor, a new memory controller that incorporates cache memory, and new Input/Output (I/O) Shop Replaceable Assemblies (SRAs) that can access memory directly without the aid or intervention of the processor. The update will increase the performance of the AN/AYK-14 by a factor of two, as well as offering weight, power, and size reductions.

The new single-card processor, as well as the memory controller module, will be capable of being backfitted into existing chassis that are now in use. In this way, a platform may take advantage of the new technology and capabilities without expending large sums of money and time to completely alter the computer system. Additionally, the new SRAs are being designed such that with a small number of modules, a new user can utilize the AN/AYK-14 as a module set to achieve its computational requirements. The new SRAs are currently planned to be available for ordering at the beginning of 1986.

Preliminary plans are being established to consider the application of the Very High Speed Integrated Circuit (VHSIC) technology to the AN/AYK-14 looking toward requirements which will exist in the late 1980's and early 1990's. It is anticipated that VHSIC will provide additional increases in speed and capability system reliability, fault tolerance and fault isolation, while reducing size, weight, and power.

Production Planning of the AN/AYK-14 Program indicated that the firm and projected requirements for AN/AYK-14 computers will exceed 10,000 through 1990. The majority of these systems will be used on many critical Navy weapon systems; e.g., F-18, LAMPs, AV-8B, E-2C, and P-3C. In view of these critical weapon system requirements, the Navy has embarked on Second Source Program for the AN/AYK-14 Computer. The second source program will be based on the "Build-to-Print" Approach using the Technical Data Packages (TDP) procured from the current AN/AYK-14 supplier. The Naval Avionics Center and the second source contractor will perform a joint validation of the TDP. Concurrent with validation the second source contractor will begin necessary efforts to establish a production capability. Production competition between the current supplier and the second source is now planned for FY 1986.

5.6.3 Support Software Development Program

In 1970 the Naval Material Command initiated an effort to develop a program generation environment, consisting of a set of standard support software tools, intended to complement the standard embedded computer hardware resources in the Navy inventory. The objectives of this initiative were to unify developments, provide centralized configuration control, avoid redundant and/or proprietary developments of support software, and insure a higher degree of support software commonality for both development and life-cycle support across many system application software areas. The achievement of these objectives will provide significant economic returns in addition to a reduction in the time and resources necessary to establish software support facilities for industrial developers, Navy laboratories and test agencies, and Navy software support activities.

Initial implementation of the standard program generation environment was focused on the then new AN/UYK-20 computer. Selection of the AN/UYK-20 was based on a forecast of a large number (over 50) of anticipated system applications and the knowledge that this forecast use of AN/UYK-20's would inherently call for a large number of support software installations. The initial estimate of development cost for a support software base was established at $8M to $15M. Using the lower side of this estimate and multiplying by the anticipated variants of the installation of the AN/UYK-20 support software base, it is easily projected that a divergent, noncompatible, independent development approach would have resulted in a cost of approximately $400M. Additionally, the life-cycle support cost of these indepen-dently developed support systems was estimated at $250K per year per system. Thus the divergent development approach for program generation environments would have presented an ongoing annual cost of $12.5M.

As history has proven, early estimates of AN/UYK-20 usage were very low as over 3250 AN/UYK-20 computers are now used in over 150 different systems. Also, the development of the support software base itself cost $15M rather than the lower $8M estimate. With an additional investment of $5M, this initial host-dependent base was revised and upgraded to a hostindependent transportable subset of ANS FORTRAN ANSI X3.9-1966, and has since been upgraded to be compatible with ANS FORTRAN ANSI X3.9-1978. This $20M investment, however, has resulted in a standard transportable program generation environment (one single set), called the Machine Transportable Support Software (MTASS) system, which is used extensively in more than 180 different facilities. This system presents an annual life-cycle support (upgrading, configuration management, version release control, etc.) cost of approximately $1M. Most importantly, this base can now be hosted by any computer facility that hosts an ANSI standard FORTRAN compiler and a minimum set of resources (in terms of available memory and file capabilities). To date, MTASS has been hosted on the following computers listed in Table 5-3.

TABLE 5-3: Systems Capable of Hosting MTASS

IBM systems	360	Honeywell systems	66
	370		600
CDC systems	6400		6000
	6600	Univac systems	1108
	6700		1110
	Cyber 70, 170		AN/UYK-7(V)
DEC systems	DEC System 10		
	DEC System 20		
	VAX 11/780		

Because iterative investments in implementation of host-dependent versions of the standard program generation environment are not needed, additional effort can be directed toward enhancing, augmenting, and upgrading this single base for the benefit of all users. A second major benefit of the transportable standard program generation environment is that new tools for software development, test, and validation need be developed only once and become immediately available to all users of that program generation environment. Thus, the total investment is significantly lower and the quality and capability of the product is significantly higher. At present, the Navy has a recorded total investment of $40M to $50M for a product improved in both functional capability and operational reliability, and showing an order-of-magnitude cost savings compared to the estimate of $400M.

A third benefit of the standard transportable program generation environment, when combined with a controlled standard instruction set architecture (ISA), was demonstrated in 1976 to 1979 when the Navy acquired the AN/AYK-14. By Navy specification the AN/AYK-14 was designed to meet the AN/UYK-20 ISA with controlled minor variants. The result was the ability to make minor compatible modifications to MTASS components to include necessary critical items in support of the AN/AYK-14. The full MTASS was provided to the first AN/-AYK-14 user system developer more than six months prior to the availability of the first preproduction AN/AYK-14 hardware. The cost of this MTASS modification was under $250K and it allowed the user system developer to complete over 80% of the AN/AYK-14 application software design and implementation prior to receipt of the AN/AYK-14 computer equipment. In the normal scenario of system development using-non standard products, the cost would have approached $8M and the schedule delay would have exceeded 18 months.

Most recently, MTASS has been extended to support the AN/UYK-44 computer, the new implementation of the Navy standard 16-bit ISA. The MTASS concept is also being applied in the development of standard support software for new hardware implementations of the Navy standard 32-bit ISA, specifically the AN/UYK-43 computer. Hereafter in this plan, MTASS/M refers to MTASS for Navy standard 16-bit ISA computers, and MTASS/L refers to MTASS for Navy standard 32-bit ISA computers.

The new DoD standard high order computer programming language (HOL), called "Ada," has been developed, and an Ada Joint Program Office (AJPO) was established by DoD in December 1980. A Deputy Director for the Navy is assigned full time to the AJPO. In close coordination with the AJPO, the Navy intends to establish Ada as the single programming language to be ultimately used in the development of all Navy embedded computer systems application software.

The implementation of a Navy standard Software Engineering Environment (SEE) is the long term goal of the Navy support software development program. The Navy standard SEE will be oriented toward Ada, and will support all aspects of Navy embedded computer systems software acquisition including requirements definition, specification, design, coding, configuration management, documentation, integration, test, validation, and life-cycle support. The definition and specification of this SEE will incorporate the following high level requirements:

a. The SEE must specifically provide for Ada HOL support and associated program generation capabilities. In addition to new software development in Ada, the SEE must support cost-effective evolutionary transition of software written in current Navy standard HOLs to Ada. All efforts related to current Navy standard languages will be in concert with this goal.

b. The SEE must be capable of supporting all present Navy standard computers, and augmentable or otherwise extendable to support future standard embedded computer hardware developments.

c. The SEE must provide for the cost-effective transition of embedded computer systems software from current to new Navy standard computers.

Near-term goals of this project include the development and implementation of host-independent Navy standard program generation environments (compilers, assemblers, linking loaders, system generators, etc.) to support the new Navy standard computer hardware developments (AN/UYK-44 and AN/UYK-43). Support for these new computers, as well as the current Navy standard computers, will ultimately be consolidated in a Navy Ada program generation environment, thereby providing systematic evolution to Ada.

The Navy Ada program generation environment will form the basis of the Navy standard SEE, and will be acquired and/or developed in consonance with the objectives of the AJPO as well as the Management Steering Committee for Embedded Computer Resources (MSC/ECR) as outlined in the Defense Computer Resources Technology Plan of June 1979.

The specific support software development activities being undertaken are as follows:

a. Augment the current standard transportable program generation environment (MTASS/M) to support the new implementation (AN/UYK-44) of the Navy standard 16-bit ISA. This development was completed in February and is now undergoing extensive testing.

b. Augment current AN/UYK-7 program generation environments to support the new implementation (AN/UYK-43) of the Navy standard 32-bit ISA.

c. As coordinated by the Ada Joint Program Office, and in conjunction with Army and Air Force Ada efforts, acquire and/or develop a baseline Navy Ada program generation environment to support the AN/UYK-44 and AN/UYK-43 computers.

d. Augment and extend the baseline Ada program generation environment to support all Navy standard computers and programming languages, resulting in a single Navy standard transportable program generation environment.

e. Define, specify, and acquire/develop a Navy standard Software Engineering Environment which will incorporate the Navy standard program generation environment, and will be based on the Ada Programming Support Environment (APSE).

5.7 Summary

DoD and the services have implemented a number of joint and independent standardization programs in order to effectively manage embedded computer resources which will help control the rapidly escalating costs of embedded computers. The thrust of these programs are directed at both software and hardware issues, each of which has life cycle cost and supportability implications. The DoD has implemented joint programs where possible, such as the Ada high order language (HOL) program. In other cases, the services have implemented separate programs to meet unique mission needs. However, these service efforts are coordinated and information is shared to ensure that cross-utilization of computer resources is considered when new weapon systems are developed.

The Congress raised several issues regarding computer standardization approaches. Specifically, the study is to include justification for development projects, issue (e), and a plan to reduce proliferation of these computers, issue (f). This section describes on-going DoD and service programs which include Ada, the Army/Air Force program for the Military Computer Family (MCF) known as Nebula, the Army's efforts at developing MCF, the Air Force program for ISAs (MIL-STD-1750A), and Navy efforts at hardware standardization through programs such as the AN/UYK-43 and AN/UYK-44--all of which explain DoDs efforts toward reducing computer proliferation.

Ada is a modern high order language which will become the standard language for writing software for DoD embedded computer applications. The Ada program extends well beyond simple language standardization and will help control the cost and improve the software quality through use of modern software development tools and state-of-the-art software engineering practices. The Ada Joint Program Office (AJPO) is managing the DoD effort to develop, implement, introduce, and provide life-cycle support for Ada.

After several years of analyzing several approaches to ISAs, the Army, in conjunction with the Air Force, embarked on a program to establish a DoD sponsored ISA program. This program, popularly known as Nebula, is focusing on developing an ISA with a 32-bit capability. The development effort is being implemented through MIL-STD-1862 and will meet the needs of evolving systems including the Military Computer Family (MCF) program.

The Army has embarked on the MCF program in order to reduce the proliferation of computer types which are embedded in weapon systems. MCF provides a family of computers, computer boards and chip sets designed to meet most form, fit and function requirements of evolving Army systems. This program will improve the Army's ability to support computers in the field.

In addition to using Nebula to meet 32-bit architecture requirements, the Air Force is implementing a 16-bit ISA standard (MIL-STD-1750A) in order to meet its requirements. The Air Force has taken this approach in order to focus on software standards while not focusing on hardware standards. Efforts toward further standardization will continue, but the primary efforts will be placed on standard interfaces.

Because the Navy has the need to support many remote weapon system platforms, hardware standardization is a requirement. This approach greatly reduces the cost and difficulty of supplying and maintaining systems afloat.

Therefore, the Navy is continuing to develop standard computers to be installed aboard ships to handle computer and signal processing requirements. Programs such as the current AN/UYK-43 and AN/UYK-44 are designed to meet future requirements and replace current AN/UYK-7 and AN/UYK-20 computers.

Software upward compatibility between current and new computers, gained through use of a standard ISA, will enable re-use of approximately $50 Million worth of Navy-standard support software. It will also enable projects having a large investment in applications software to transition to new, high-technology computers with minimal rewrite of that software. The Navy has a total investment of approximately 100 million source lines of applications software which is unique to Navy standard ISAs. As much as half of this will be transitioned to the new AN/UYK-43 and AN/UYK-44 computers.

SECTION 6 PLAN FOR PROLIFERATION REDUCTION

6.1 Introduction

This section presents the plan for reduction of proliferation requested by Congress and which has been discussed in the previous sections of this report. This plan is believed to be a rational management approach which necessarily balances three forces: competition, technology insertion, and proliferation reduction for logistics and maintenance reasons. During the period of transition to this plan, open communication among Congress, Industry, and DoD may be considered of utmost importance. The general direction of change must be agreed at the outset so unnecessary divergence is avoided. With the general direction established, more detailed implementation plans with schedules and associated costs can be prepared.

The Senate Armed Services Committee report which accompanied the FY1983 Defense Authorization Act clarified the intent of Congress in enacting Section 2315 of Title 10, United States Code, normally referred to as the Warner Amendment. This legislation excluded procurement of certain ADP equipment and services from the provisions of Public Law 89-306. The equipment and systems which have been excluded are commonly referred to as "Mission-Critical Computer Resources (MCCR)". These exclusions have been viewed as an incentive to seek improved and streamlined methods and practices for procurement of systems. DoD has issued guidelines and provided further definition relative to managing mission-critical computer resources. This plan first briefly mentions the Non-Mission-Critical Computers and then concentrates on proliferation reduction in mission-critical computers and systems.

6.2 Non-Mission-Critical Computers and Systems

For computers and systems which are not critical to the military mission, normally referred to as business or management information systems, Ada may be considered a candidate language just as COBOL and FORTRAN have been. If Ada is used, then a validated Ada compiler implementing MIL-STD-1815 will be required. DoD has no intention of controlling Instruction Set Architectures or hardware for these applications; normal requirements and procurement processes should apply.

Where commercial products can be used without sacrificing defense capability or incurring unreasonable costs for maintenance, logistics and software, it is DoD policy to use them (cf. DoD Directive 5000.37). There is nothing, however, to prevent commercial computer manufacturers from implementing any of DoD's computer functional interface specifications in commercial products if they so choose. In fact, it is anticipated that many of these functional interface specifications will be implemented, by choice, in the commercial sector for reasons of performance, economy, competition, and the need for software-compatible computers.

6.3 Mission-Critical Systems

Mission-Critical Computer Resources will be discussed in two parts. Section 6.3.1, Mil-Spec Environment, includes those applications which cannot use commercial hardware without modification because of the environment. Section 6.3.2 Non-Mil-Spec Environment, covers those applications for which the environment does not rule out the use of commercial equipment, but other factors may.

6.3.1 Mil-Spec Environment

Current trends toward more extensive use of embedded computers in defense systems are expected to continue. Likewise, there is every indication that proliferation will continue unless appropriate management directions are given. Without proliferation reduction, the capabilities of the armed forces will be significantly reduced from that otherwise attainable.

Systems organic to and in direct support of the fighting units, i.e., warfare systems which must function reliably in worldwide environments under wartime conditions and be supported organically, demand standardization. When a computer failure occurs, the capability must exist to restore system operation through local maintenance procedures, automatic transfer of function to another computer, or interchanging computers from less critical systems and functions. With increased use of computers in networks and in distributed systems, object code transportability has become essential. The ability to run the same software at nodes of like function reduces the amount of software to be developed and maintained and increases the survivability of automated operational functions. In addition, even with the high MTBF of new miniaturized computers (approaching 10,000 hours for large "mainframe" computers), failures can be expected during the long periods (90 days) associated with battlefield or naval missions. Even 10,000 hours, MTBF, allows a 20% probability of failure on a 90-day mission. This means that remote operating units must carry an inventory of spare parts with them. If each Service were required to support a proliferation of 40 computer types in the field,the cost of organic spares alone would be over a billion dollars which is deemed to be unacceptable. Note that this does not include spares required to react to battle damage during hostilities.

Needless proliferation of hardware and software components is the root cause of the problems elaborated in the earlier sections of this study. Technology alone will provide only partial amelioration. Intelligent, evolving approaches to reducing proliferation that deal properly with competition and technology are the only known solution. Specifying standards and controlling their evolution provides substantial economic and operational benefits to the Department of Defense and, at the same time, is compatible with the need to maintain a strong industrial technology and support base that can be expanded rapidly for mobilization. The following sections describe the plan for controlling proliferation while using new technology at several levels--High Order Language, Programming Support Environment, Instruction Set Architecture, and Computer Hardware.

6.3.1.1 High Order Language (HOL) Standardization

There is general agreement that standardization at the HOL level is necessary to keep software costs in check. Analyses conducted during the period 1974-1976 forecast cumulative cost avoidances of as much as $24B over 30 years derivable from HOL standardization with respect to the use of embedded computers in DoD[20]. In 1974 it was conjectured that HOL needs were not significantly different across the three Services and that a common HOL might be feasible. Subsequently, this conjecture was validated through the joint development of a detailed requirement for such a common language. The Ada Program was DoD's direct response to this opportunity. Ada is now published as MIL-STD-1815A and approved as an American National Standards Institute (ANSI) standard. As discussed in Section 5, standardizing at the HOL level alone does not solve the problem of transportability of HOL software. The conclusion of the present study is that control of the HOL specification is necessary but not sufficient. Additional standards are necessary as discussed below.

OSD has elevated DODI 5000.31 from an Instruction to a Directive, DODD 5000.31, and directed that, effective 1 January 1984 all programs entering Advanced Development (R&D budget category 6.3) must have plans to use Ada for implementation. In addition, all programs entering Engineering Development (R&D budget category 6.4) after 1 July. 1984 must have plans to use Ada. Initially, Ada will co-exist with other languages, particularly COBOL, FORTRAN, CMS-2, JOVIAL and many assembly languages, until systems programmed in these languages are phased out or it is cost effective to rewrite them in Ada, perhaps as part of a major modification program. Ada is expected to have a primary life of about 25 years. Subsequent HOLs must initially co-exist with Ada. After approximately 20 years, Ada is anticipated to be phased out naturally as new languages appear to replace it but, like FORTRAN and COBOL, can be expected to be productive for a long time to come. These time estimates may change depending on technology evolution, but are today's best estimates.

6.3.1.2 Programming Support Environment (PSE)

For software development support for Mil-Spec computers, the DoD plan is to continue initiatives begun under the Ada Program. As a part of the Ada program, the Army and Air Force have undertaken development of transportable PSEs which will run on a number of commercial computers (rehostable) and will generate code for a number of types of Mil-Spec computers (retargetable). These PSEs are the Army Ada Language System (ALS) and the Air Force Ada Integrated Environment (AIE). The Navy Ada Language System (ALS/N) will build upon the Army Ada Language System and it is planned that the two systems will be phased together under common configuration management as the Army/Navy Ada Language System.

DoD will encourage hosting of the ALS and AIE on a wide variety of commercial hardware. The intent is to take maximum advantage of existing and future commercial host computers and operating systems for software development and life-cycle support. Rehostability provides for reducing the cost of moving the PSE to new and different commercial host computers and operating systems and thereby prevents needless proliferation of PSEs. Retargeting provides the means for accommodating new Instruction Set Architectures (ISA) when necessary because of mission requirements or a break-through in ISA technology.

PSE, utilizing a common interface set available to industry, with configuration management of interfaces between the PSE and software engineering tools, is seen as a major step toward transition of new software technology from the lab to widespread use in practice more quickly than now feasible. Further, it removes some impediments to "re-use" of applications software, provides a base from which data exchange (interoperability) between tactical systems can be improved, and provides for eventual back-up (continuity of operations) between software development and support centers. As with Ada, it is anticipated that this PSE will not be static, but rather will be managed so that it evolves with new technology and the availability of new tools. Concentration of resources toward a common PSE is expected to improve return on investment as compared to supporting many approaches.

A common PSE offers greater opportunity for improving software productivity while achieving greater reliability and adaptability of embedded computer systems than possible with just a common HOL. Toward this end, OSD has planned a new software initiative--Software Technology for Adaptable, Reliable Systems (STARS) to improve the state of practice of software engineering for defense systems by improving the skills, software tools, and management practices that constitute the total environment in which software is developed and supported. The objectives of the STARS program are to:

(1) Improve the personnel resources by increasing the level of expertise and expanding the base of expertise available to DoD,

(2) Improve the power of the PSE by providing better project management tools, application-independent technical tools, and application-specific tools, and

(3) Increase the use of software tools in the PSE by improving business practices and usability, and by increasing the levels of integration and automation.

The initial approach will be to use specific PSEs developed either under government sponsorship or independently by industry in accordance with general DoD requirements. The long-term approach will take advantage of work done on new tools (without having to convert code to another PSE) through use of an extensible core environment with standard interfaces. This effort will build on the Ada Programming Support Environment (APSE) work now in progress. As the common interface specifications for environments are managed over time, and as new tools and methodologies are available (whether government-sponsored or not), they will be immediately available to the full range of government and industry. This is the means for inserting technology into PSEs.

Adequate testing of Ada PSEs will be accomplished prior to making their use mandatory. The current schedule forecast is initial availability of validated Ada compilers during late FY83 and FY84. (Three compilers have been validated as of the date of this report). During FY1984 and 85 these compilers will be tested along with operational code produced by them using the concept of Beta test sites. During this period, initial environments will be rehosted and code generators targeted to DoD Instruction Set Architectures will become available. By 1986 Ada should be in widespread use throughout DoD in the development and upgrade of its systems at a production quality level.

6.3.1.3 Instruction Set Architecture (ISA)

The Instruction Set Architecture (ISA) is the interface between software and the hardware that will execute it. Computers of different size, technology, or manufacturers that will execute the same object code software are generally termed "software compatible" or "object code compatible" computers. In general, each commercial computer manufacturer retains its own and its customers' investment in large-scale software systems at the ISA level as well as the HOL or PSE level. This is why major industry computer users remain with the same supplier, the reason that commercial computer manufacturers produce ISA-compatible lines over time and the reason that a sub-industry has developed to produce computers having the same ISAs as the computers produced by leading computer manufacturers. If software could be transported so easily at the HOL level as some have claimed, then this sub-industry would have no economic basis on which to exist. Software profitability depends upon specifications for HOL, PSE, and the ISA. They contribute major savings in software/-hardware integration costs.

Since object code compatible computers based on vendor-owned and legally protected ISAs are essentially sole-source and limit competition, the Services have experienced problems in using them. Each service has made a concerted effort to force the problems to a manageable number by converging toward vendor-independent ISAs which enhances hardware competition. This convergence has taken place over the last 15 years, essentially independently in each Service. Currently, the Army has developed and the Army and Air Force are jointly controlling and refining NEBULA, the 32-bit ISA specified in MIL-STD-1862B. The Air Force has had good success with its vendor-independent, 16-bit ISA for avionics--MIL-STD-1750A. The Navy has been using two standards--those of the AN/UYK-20 and -7--and has recently moved to two upward-compatible ISAs--the 16-bit ISA of the AN/UYK-44 computer and the 32-bit ISA of the AN/-UYK-43 computer.

This initial convergence is consistent with goals to provide the most cost-effective defense capabilities. The Department considers the joint use and joint control of NEBULA by the Army and the Air Force to be a positive step. No technical or economic evidence has been found that these approaches should be changed in the near term. What is clear is that vendor-independent ISAs are necessary to maintain maximum access to competition. Although common ISAs represent some change and will require a transition period before being fully effective, they represent a significant improvement for the Services.

Object code compatibility for target computers is the most significant factor in re-use of support software. A significant amount of the PSE (support software) depends on the ISA of the target computers. Examples are compilers, linkers, loaders, simulators, debuggers, etc. Introducing new ISAs incurs additional expenses associated with these tools and will, in all probability, result in some incompatibilities. The approach to the PSE stated above will, however, minimize these costs when new ISAs are necessary.

The ability to transport embedded computer software, even though developed in Ada, to a battlefield computer having an ISA different from that of the original battlefield computer will be impeded by dependencies of the original software on the ISA of the original computer. Consideration must of course, also be given to software interfaces and to underlying support (run-time operating system, hardware interconnects, etc.).

At the current state of knowledge, art and practice, there are always some ISA dependencies. This can be easily shown in the commercial market place as well and is a major underlying factor in the current Personal Computer shake-out, wherein home computer buyers are demanding that their new computers be able to use existing software.

The cost of moving software from target ISA to a different target ISA is dependent upon several variable factors and cannot be estimated precisely. Current estimates range from 10% to 100%. One major contributing factor is whether the software is written in a target-computer-unique HOL (or assembly language) or a target-independent HOL. Other factors include the extent of re-testing required, and the extent to which the manner of packaging or binding (putting together the various individual programs making up the software) is dependent upon ISA peculiarities (i.e. memory management, addressing limitations, etc). The point is, however, that even with a target-independent HOL (such as Ada), the cost of moving from one ISA to another will be at least 10% of the original software development cost for retesting, even if there are no ISA dependencies in the common HOL. The cost of moving software written in assembly, or a target-unique HOL from one ISA to another will range from 50% to 100% of the original development cost.

High Order Languages and Programming Support Environments are used in the development and support of these systems; they are not used on the battlefield. Object-code-compatible, Mil-Spec computers for the battlefield offer several advantages to the services with respect to their combat systems. Computers are being used in networks of systems to exchange necessary information. Object-code compatibility gives the capability to transfer the processing load between computers within networks without creating additional software to be maintained. When the necessary physical interfaces are added, object code compatibility allows computers to be interchanged on the battlefield to keep the most important systems running by using a computer from a less critical system.

During system development, use of object-code-compatible computers allows software development to begin on current hardware (or emulators), while newer hardware is being developed for the system to be fielded. This reduces both time and risk to field new systems. For systems already fielded, it allows for inserting new hardware technology while retaining operational software. Further, it allows ISA-dependent operational software from earlier systems to be transported directly to newer hardware, or portions of the older software to be combined with new software under development. This is the case with the current Navy AN/UYK-43, -44, and AN/AYK-14 computers and, in industry, with the 8085, Z80, 8086 and other families.

The current DoD ISA standards are within the technology generation of commercial counterparts and, like the latter, have been, are, and will continue to be dynamic standards. Evolution and control mechanisms are in place for each of the current DoD ISAs and improvements that support DoD needs or that offer cost or efficiency benefits will be made over time. At this point, there is no clearly-preferred future ISA direction. Some researchers see ISAs moving to a higher level, closer to the HOL (including more of the PSE in hardware), employing more complex and more powerful instructions (language-oriented computers). Although technically attractive in an abstract, one must consider the end-to-end efficiencies involved in a so-called language machine. It is not at all certain that there will be a net benefit, especially in real-time systems. As evidence of the lack of potential benefits, we have not seen any serious commercial applications of high level language computers. Therefore other researchers believe this approach to be inefficient and in the wrong direction. The latter are working toward Reduced Instruction

Set Computers (RISC)--the use of very simple instruction sets that can serve as primitive building blocks.[19] Research is also in progress on highly parallel instruction sets that do not follow the traditional Von Neumann model, but are data and demand driven, e.g., data-flow and reduction architectures.[19] Language-oriented or RISC computers may offer advantages over current ISA specifications. However, each of these approaches also has an internal ISA, and while it may be at a level somewhat higher or lower than that of more traditional and current ISAs, it remains simply yet another ISA with all of the attendant language dependency weaknesses, as discussed in Section 5.

Based on arguments from both sides, software and hardware, it is concluded that object code transportability, hence ISA standardization, is essential to improve competition, to reduce the costs associated with the computer life-cycle and to provide for acceptable survivability and continuity of operations of combat systems. Moreover, some mechanism for object code compatibility will remain essential until improvements in the state-of-the-art permit otherwise. The programs established in each Service to satisfy growing mission-critical computer needs should not be interrupted now in an attempt to achieve a greater rate of convergence. For example, immediate Navy adoption of NEBULA would mean cessation of the current (and nearly complete) efforts to succeed the aging AN/UYK-20's and AN/UYK-7's on its ships by newer, but software-compatible computers. This would fail to take advantage of the existing, extensive Navy PSE and would leave the Navy with a gap of several more years without an Ada or CMS-2 environment for immediate use. Similarly, it would not be wise for the Air Force to stop its MIL-STD-1750A Program, for which there now exist over 20 highly-competitive sources for airborne computers and several sources of PSEs. On the other hand, if the Army and Air Force were to drop NEBULA (MIL-STD 1862) efforts, then the DoD community would lose a very advanced and powerful ISA for future 32-bit applications, and the Army would undergo increasing battlefield computer proliferation with attendant logistics problems for a number of additional years.

DoD does not plan to retain current ISA standards indefinitely. It will continue to monitor and sponsor industrial and academic research efforts in ISAs as part of its technology base program in order to establish a direction for its future ISAs. The goals for future Mil-Spec computers after the current generation and the tentative timetable are discussed below in Section 6.3.1.6.

6.3.1.4 Hardware

From the above, it can be recognized that HOL, PSE, and ISA commonality all contribute to reducing software development and maintenance costs. Reducing proliferation of ISAs is necessary for hardware proliferation reduction and non-proprietary ISAs are necessary for competitive acquisition of object-code-compatible and -interchangeable target computers for the battlefield. Yet, none of these specifications actually reduces hardware proliferation.

In many cases, reducing hardware proliferation is essential to ensuring combat survivability, improving readiness through common spares, and reducing maintenance and training costs. Since hardware standardization does have the effect of limiting competition and restraining technology, DoD resorts to it only when necessary.

Embedded computers are essential to the survivability and mission effectiveness of modern weapon systems. Examples of combat functions which require operating computers to perform their mission are point defense, target acquisition, fire control, and damage control. A weapon system that loses any of these capabilities in combat cannot perform its mission, and may be in imminent danger of being destroyed. To prevent destruction and restore mission effectiveness, it is essential to get the computer operating again immediately. This requires a swap-out of the defective or damaged computer. Fault diagnosis and repair, no matter how trivial, simply takes too long in the heat of battle. Swap-out requires either carrying complete spare computers, which is impractical for forward units, or the ability to remove operating computers from less critical functions and substitute them into essential functions. In Navy shipboard design, the swap-out is usually done with pre-installed cabling and switching arrangements. This is already done for other items of equipment such as radios. Battlefield substitution of entire computers requires hardware standardization at, at least, the form, fit, and function (F^3) level.

For meaningful deterrence and adequate war-fighting capability, weapons systems must be available when needed. Operational availability depends not only on the mean time between failure (MTBF), but also on the mean time to repair (MTTR). For remotely deployed units, the primary determinant of mean time to repair is the availability of deployed spares. The time required to obtain spares from a depot or home unit is intolerable.

Deployed units must therefore carry spares for each of their essential pieces of equipment. As shown in Section 2, provision of adequate spares is a major factor in life-cycle costs. Reducing life-cycle costs therefore requires reducing the number of required spares. Also as shown in Section 2, the requirement for spares increases sharply as the number of different types of equipment to be supported increases, but increases only slowly (and eventually levels off) as the number of multiple copies of the same type of equipment increases. Providing contingency spares for a multiplicity of different computer types increases the probability that some of the spares will never actually be used, but will merely carried along as an expensive and wasteful form of insurance. Holding down life-cycle costs therefore demands hardware standardization at, at least, the logistically-identical level.

Some observers have maintained that advancing technology will solve the spares problem by greatly increasing equipment reliability. These observers fail to note that, as concluded in Section 5, advancing technology offers two types of benefits:

(1) The potential to meet established system requirements with computers that are more cost-effective and have improved physical parameters, e.g., size, weight, power, reliability, etc.; and

(2) The potential to increase system capabilities to meet new threats to our national defense.

Those who suggest that technology will solve DOD's logistics support and operational availability/survivability problems are considering only the first benefit listed above. They are also ignoring the possibility of battle damage and the necessity for war reserves. The failure-free device loses some of its appeal when this factor is considered, that is when the MTBF is much greater than the expected mean time between battle damage events. If future mission requirements are no different than current requirements, then technology would provide a gradual reduction of the number of logistics items to be supported in future embedded computers. Such a hypothesis, however, is not supported by history. To the contrary, we have seen a constantly increasing growth in mission requirements and a corresponding growth in the capabilities of embedded computers. The discussion is reduced to an argument of whether or not growing technological capabilities will exceed growing needs. If capabilities exceed needs, then the number of logistics support items will decrease as mission requirements grow, but if not, then the number of logistics support items will either remain constant or increase.

Some embedded computers have become physically smaller and the number of their logistics items per computer has decreased. However, not all new computers are reduced in size relative to their predecessors. Several of the newer off-the-shelf Mil-Spec computers, for example, are actually larger than their predecessors. Obviously, producers of these computers chose to increase performance and capacity rather than to keep such parameters constant and reduce size or improve reliability.

As noted above, hardware standardization can have the effect of limiting competition and restraining technology. Therefore, DoD resorts to hardware standardization only when necessary. Situations that obviate the need for hardware standardization are:

(1) the end-item is so small or inexpensive that it is not economically repairable, and

(2) centralized maintenance is feasible, even in deployed units.

An example of an end-item that is so small or inexpensive that it is not repairable is the small air-to-air or air-to-ground missile or rocket. This application uses large quantities of small, relatively inexpensive computers that are usually based on a microprocessor, often have a circular form factor, and are seldom procured as a separate end-item. The DoD does not intend to standardize the hardware for small missiles, but does use a standard HOL and will use a standard ISA such as MIL-STD-1750A as soon as microprocessors implementing that standard become available. Other considerations of non-standard hardware are below in paragraph 6.3.2.3.

There are two DoD programs specifically aimed at developing computer hardware for applications requiring common spares. These are the Navy's AN/UYK-43 and -44 and the Army's Military Computer Family (MCF) programs. These are explained in detail in Section 5.

The Navy has competitively selected one producer for the AN/UYK-43 and -44, based on fixed-price competition for five years of production each. These new computers are now available for, and are in use in, development systems. The AN/UYK-44 is in use in one system which is being prepared for at-sea testing, and the Navy will take delivery of the first AN/AYK-43 for the DDG-51 shipbuilding program in September 1983. The AN/UYK-44 will be in full scale production by September, 1984 and the AN/-UYK-43 by December, 1984. The Navy has acquired full engineering data packages to support alternate sourcing after the first five years, if that should become necessary.

The Army has successfully tested Advanced Development prototype MCF computers from three competitive contractors--GE/TRW, RCA, and Raytheon. With the concurrence of Congress, the competitive Full Scale Development (FSD) phase between the two winners will begin in early FY1984 with FSD models being available for test in FY1985. In early 1982, the Army conducted an analysis of 134 current and planned battlefield systems which use computers of the size and type of the Military Computer Family (MCF). Seventy-four were considered too far along to use the Ada Language System (ALS) and MCF initially. Analysis of the remainder indicated production quantities which would support, cost-effectively, two producers. The remainder included systems such as the Advanced Field Artillery Tactical Data System (AFATDS), which has issued a Request For Proposal for advanced development. It and other systems in the planning stage will develop plans for transitioning to ALS and MCF for decision at appropriate milestones. The leader-follower approach is believed to be most effective at present. If technology progresses faster than expected and if it is cost effective, then the possibility of competition at the box or module level on a form, fit, and function (F^3) basis is still available in the Army approach. This decision can be made at any time prior to production.

Use of MCF does not imply that a given system will use only Mil-Spec MCF computers. As less costly, software-compatible MIL-STD-1862 machines become available, they are expected to be used in the appropriate environments and applications. Likewise, MCF will not solve all requirements such as signal processing and low-end applications.

Additionally, the Army expects software-compatible, less-expensive, less-than-Mil-Spec NEBULA machines to become available. This would be similar to the case of competitively-available MIL-STD-1750 machines. The goals of future Mil-Spec computers are given in Section 6.3.1.6.

At the current state of the art, the ability to interchange internal modules (spares) made by different manufacturers totally on an F^3 basis has not been demonstrated. Therefore, both the Army and Navy programs are based on common hardware. It is hoped that the succeeding phase of Army and Navy computer production will not have that constraint as discussed in section plan below.

Both of these developments use the latest semiconductor technology. The method of technology insertion into the AN/UYK-43/44, AN/AYK-14, and the MCF production phase will be by Pre-Planned Product Improvement (P^3I) and by Engineering Change Proposals (ECPs). Approved changes will be applied to all affected computers and spares in order to provide logistics commonality, survivability, and operational availability. The most economic production phase for these current programs has been estimated to be between four and eight years.[19]

The findings of this study indicate that current hardware development efforts to implement DoD ISAs into hardware should be continued for the near term. The specifications implemented in this hardware generate natural F^3 specifications for succeeding generations in which more competition can be gained and emerging VHSIC technology can be used to advantage without software loss.

6.3.1.5 <u>Plan for Reducing Proliferation, Encouraging Competition, and Using New Technology in Embedded Computer Resources</u>

The direction toward Ada and validation of Ada compilers is straightforward. All Services and OSD are committed to transition to Ada and are initiating efforts in the Software Technology for Adaptable, Reliable Systems (STARS) program to build on the base established in the Ada program.

DoD and the Services are committed to the STARS program to improve the programming support environment (PSE) to build improved combat systems while economizing in manpower resources for development and maintenance of the large amounts of software yet to come. To this end, DoD, the Services and industry are actively cooperating to develop a common set of interface specifications so tools can be inserted in the Ada Programming Support Environments (APSEs). The direction is to provide a framework for allowing new technology to be inserted rapidly and then distributed widely to reduce the time between development of new technology and its wide-spread use. DoD and the Services are working with industry to define the best paths and remove obstacles. Software proliferation reduction will be through a common, rehostable, retargetable core environment into which new technology can be inserted. The PSEs, based on Ada, are expected to mature over the next five years and Ada should be sound for 20 to 25 years, if past experience with languages is an indication.

The most difficult problem is the solution to hardware proliferation reduction. It is clear that total standardization of hardware across DoD cannot be achieved and should not be attempted.

There are several alternatives for the future:

(1) Continue the state of proliferation, pay the bill, and accept the reduced operational availability/survivability caused by it;

(2) Continue the state of proliferation and pay a higher bill to enhance survivability in the face of proliferation by designing in backup capabilities based on incompatible computers;

(3) Reduce proliferation through individual Service standardization efforts;

(4) Reduce proliferation through DOD-wide standardization.

The first alternative is unacceptable. Reduced operational availability and survivability should not be tolerated, when more cost-effective alternatives are available that provide the desired capability. In fact, to meet mission requirements, survivability and operational availability must increase. The second alternative is indicative of the past and is not acceptable because of increased resources needed to achieve the same capability. Consciously accepting a higher cost to achieve the same capability is not considered wise.

Alternative three represents the current state of practice in DoD. Current programs are based on numerous studies and past experience. It will be approximately five years before all products of the current programs are in the field. During this time further analysis will necessarily be conducted to determine the best course for OSD and the Services to take for the next generation and whether alternative four is the proper direction.

It is safe to predict that DoD will eventually move away from existing ISAs as a function of the benefits offered by new ISAs and the costs of developing, introducing, and supporting new PSEs. The primary useful life of an ISA is expected to be less than that of an HOL, perhaps 15-20 years, versus about 25 years for the HOL because of the rapid changes in semiconductor technology. The introduction of new ISAs adopted by DoD will overlap current ISAs until the older ones are phased out of the hardware inventory. Embedded computers with new and old ISAs will coexist until it becomes feasible and cost-effective to completely replace the latter.

DoD cannot now predict when, if ever, vendor-independent ISA specifications will cease to be essential, for they provide for hardware competition and a means to use new device technology with major improvements in software transportability. If breakthroughs occur, then new vendor-independent ISAs can be added as pointed out in the PSE discussion. Whether DoD Instructions are published or not is immaterial; what is important is that vendor-independent ISA specifications for tactical equipment should be acknowledged as the way to do business. Industry must participate, and if better ISAs can be developed both to meet DoD needs and to foster competition, they will be investigated and adopted as appropriate. While DoD will work toward fewer ISAs, the current approaches will not be fully implemented for another three or four years.

Given the current state of VHSIC, MIL-STD-1750 computers, NEBULA computers, and the AN/UYK-43 and -44, continued planning for hardware evolution must occur. The development phase of a Mil-Spec computer including testing takes about 3-4 years to a production decision and an additional year to first items. The next phase of ISA analysis would have to be completed by 1987, to have the next generation of hardware ready for the early 1990s. At this stage, whether additional convergence should take place cannot be determined. However, DoD and the services have mapped out a plan and goals for the next generation of Mil-Spec computers.

6.3.1.6 Goals for Future DoD Tri-Service Mil-Spec Computers after the Current Generation

There is some question about the early attainability of hardware which can meet the operational needs of the Services while maintaining equitable competition. Whether it is possible to move away from current practice which involves language, ISA and other hardware interfaces and design considerations as well as logistics supportability remains to be seen.

The Services and OSD agree, however, that a program must be outlined and undertaken which encompasses the following goals:

(1) Ability to reuse/transport contemporary application software at the object code level within the Services as well as among the Services.

(2) Upward compatibility of the next generation computer with application software object code of the preceding standard object code of any of the Services.

(3) A computer which is an efficient target for Ada source code. Since ISAs can be implemented with different hardware technologies, the whole spectrum of ideas ranging from language machines to RISC will be considered.

(4) A modular, building block family having a spectrum of performance capabilities with each member having object code compatibility described in 1, 2, and 3 above.

(5) Other than the software transportability constraints listed above and a certain throughput range, each module of the new family shall be specified only at the external form, fit, and function (F^3) level.

(6) Sufficient overall reliability will be built into each family member through fault tolerant, self-healing, and redundant circuitry that the modules will be expected to perform for a normal 15-20 year service life time prior to degrading below a predetermined threshold of acceptable performance. Thus, organizational maintenance and organizational spares will not be required. (Note that this does not preclude the need for sufficient spares to handle battle damage as appropriate to the maintenance concept adopted.)

(7) Each member of the family of advanced computers must be available from multiple competitive sources. The implication is that industry must agree on the interface standards specified on an F^3 or other basis.

As an initial step toward items (1) through (2) above, OSD will monitor and guide Service efforts such as the Air Force High-Level Language Computer Technology Program, the Navy-sponsored Reduced Instruction Set Computer investigation, and the Navy next-generation ISA study. The objective is convergence on a common approach to DoD software reusability/transportability much as has been done for VHSIC and Ada and is planned for STARS. Participation by the computer, electronics and software industries and the impact on final technical approach will be through specific contracts, DoD/Service sponsored workshops, and requests to industry associations to nominate approaches and to appraise progress.

Additional research will also be initiated to help clearly define the characteristics of Ada "optimization." The desired goal of this research will be the perfection of direct Ada execution high-level language machines because, if practical, such architectures would allow full transport of Ada at the source code level. That is not possible now. In fact, the viability of such an architecture is far from clear. Ada optimization will be defined within the total context of life-cycle hardware and software costs as well as computer cost and performance. It is expected that, based on enhancing current research efforts, figures of merit and criteria for evaluating industry candidate Ada architectures can be developed in FY1985 for evaluation in FY1986-87.

It is expected that many candidate architectures can be adequately evaluated through simulation on supercomputers running standard DoD Ada code benchmark programs. Engineering development may be expected to begin in late 1986 to early 1987 with limited production to start in the early 1990s, if successful. Because of the lifetime failure-free goal, field maintenance, repair and spare parts requirements would be eliminated. F^3 computer modules can be concurrently and competitively provided by a number of manufacturers using their own internal electronics technology and, within limits yet to be determined, their own "architecture." The requirement for object-code transportability would not be relaxed. Organizational computer sparing for expected battle damage conditions would be at the F^3 module level.

The Navy currently has programmed advanced development funds (6.3) in the Five Year Defense Program to support the above technology investigations and to develop guidance for the engineering development phase.

All of the Services are experiencing a proliferation of small embedded microprocessors in performance and size ranges where no DoD standards exist. Because of the growth in application of these microprocessors and the rapid growth of their memory capabilities, it is reasonable to expect that the ten year DoD investment in associated applications and support software will equal the investment in DoD embedded minicomputer and mainframe software. It appears prudent, therefore, to initiate a program for a "standardized" high technology embeddable microprocessor to bridge the gap until the next generation modular family comes on line. The Navy is exploring such a program which would be based on industry-wide competition for a small family of single-board microcomputers on a limited number of form-factor boards and using a multiple-sourced, high-technology, commercial microprocessor chip set (preferably 32-bit). It is expected that a number of manufacturers could currently provide these microprocessors with the only technology constraints being board form, fit and function and object code compatibility of the multiple-sourced chip set. The Navy expects to start this program in mid FY1984 if the required reprogramming of funds can be supported. If the reprogramming can be accomplished in FY84 and 85 it is expected that a competitive full scale engineering development contract can be awarded in July 1985 and limited production will commence September 1986. Fiscal year 84 and 85 R&D funds reprogramming will be required. OSD will encourage the required reprogramming and will prepare appropriate POM 86 issue papers. In the interim, a multi-activity requirements and specifications team is being formed. Army and Air Force participation on the team is being invited and OSD will encourage their full participation.

6.3.2 Non-Mil-Spec Environments

Within mission-critical systems there are some applications which can use commercial products directly and create an area of overlap with non-mission-critical systems. Examples are ground-based fixed radars; command, control, and communications systems at fixed installations; commercial host computers for software development and support; simulators; etc. These systems generally use commercial power and are in fixed, air-conditioned facilities. Decisions to impose DoD standards in this area will be made on a case-by-case basis depending on the situation at the time.

6.3.2.1 High Order Language (HOL)

It is planned to move to Ada as the single programming language for mission-critical systems. Implementation of this decision will necessarily be governed by availability of commercial products and Ada-trained programmers.

This differs from non-mission-critical applications where Ada will be encouraged but not mandated.

The quality and availability of commercial programming environments will be a key determinant in the decisions to be made.

6.3.2.2 Instruction Set Architecture (ISA)

DoD will not enforce ISA standardization where commercial products can be used more cost effectively and where adequate competition exists.

6.3.2.3 Hardware

For mission-critical systems that can use commercial computers directly and for which the government does not have to supply parts or train maintainers through its supply system, hardware standardization will not be enforced.

DoD believes that once chips become available for its MIL-STD-1750A, MIL-STD-1862B, and the AN/UYK-43,-44 Instruction Set Architectures (ISA), then suitable commercial quality computers using these ISAs will be built competitively for these less-rugged environments, reducing support costs for PSEs, increasing software transportability between manufacturers hardware, and reducing hardware costs.

Across the Services, there are many applications particularly in fixed plant, training, and even in less critical functions on the battlefield, where less than full Mil-Spec, less costly computers can be used. Examples were given is Section 2. In these physical environments and applications, software compatibility at the ISA level would provide for hardware competition and save maintenance of additional software baselines. In those cases where hardware maintenance and training are not a problem for DoD, then common hardware need not be enforced. For those applications where logistics and training are a problem and operational availability must be attained, then a large quantity of one type of computer can be procured to appropriate specifications and used in many applications. ISA compatibility between different vendors' hardware for those instances where logistics are not important, would provide a larger base of object-code compatible machine producers from which future Mil-Spec requirements can be satisfied. Their semiconductor technology can be immediately inserted because there is no logistics impact.

6.4 Conclusion

It is concluded that in the nearer time frame national defense is best served through DoD support of Ada, the STARS program, and the government-sponsored MIL-STD-1862, MIL-STD-1750, and Navy standard ISAs. These standards are not rigidly fixed and will be modified through industry participation and use of Ada, the PSEs, and ISAs. For the next generation of military computers, DoD and the Services will work together with industry researching new Adacompatible computer architectures, and technology to eliminate or lessen the logistics requirements which today mandate hardware standardization in many applications. The approach for the next generation will allow maximum competition with concurrent embodiment of new technology.

REFERENCES

1 Currie, M. R., "DoD High Order Programming Language" (Memorandum), Defense Research and Engineering, Jan. 1975

2 DeRoze, B. C., "An Introspective Analysis of DoD Weapon System Software Management", Defense Management Journal, Vol. 2, No. 4, Oct. 1975, pp. 2-7

3 Whitaker, W. A., "The U.S. Department of Defense Common High Order Language Effort", ACM SIG PLAN Notices, Feb. 1977

4 Fisher, D.A., "DoD's Common Programming Language Effort", IEEE Computer, Vol. II, No. 3, Mar. 1978, pp. 24-33

5 Bennett, D. A., Kornman, B. D., and Wilson, J. R., "Hidden Costs in Ada", ACM Ada Letters, SIGPLAN Technical Committee on Ada (AdaTECH), May 1982, pp. 9-20

6 Nissen, J. C. D., Wichmann, B. A., et al, "Ada Compiler Specification and Selection" (Draft Report), National Physical Laboratory, 25 pp.

7 Wulf, W. A., "A View of Language Standardization", unpublished note.

8 Burr, W. E., Coleman, A.H., Smith, W. R., et al, "Final Report, Computer Family Architecture Selection Committee", Summary and VOL I - IX, ECOM-4525 to ECOM-4533, U.S. Army Electronic Command, September 1977.

9 Weinstein, W. W., and Hall, E.C., "MCF/CFA Program Evaluation Final Report", R-1300, The C.S. Draper Laboratory, August 1979.

10 Fuller, S.H., Shamman, P., et al, "Evaluation of Computer Architectures via Test Programs", AFIPS Conf. Proc., National Computer Conference, Vol. 46, 1977.

11 Fuller, S.H., Matthews, G., and Szewerenko, L., "Phase II Comparative Evaluation of the MCF Computer Architectures", Carnegie-Mellon University, January 15, 1978.

12 Dietz, W.B., Stone, H.S., and Szewerenko, L., "Computer Family Architecture Selection, Phase III Final Report", US Army CORADCOM, March 15, 1979.

13 "Comparative Evaluation of the Nebula Architecture", TR-MCF-007, EG&G Inc., March 4, 1981.

14 Kogge, P., and Olsen, P.F., "The Army's MIL-STD-1862 and the Military Computer Family", IBM Technical Directions, Vol. 7, No. 2, Summer 1981.

15 Anderson, P.G., "A Comparison of Nebula and Alternative Computer Architectures via Selected Ada Programs", Rochester Institute of Technology, December 19, 1981.

16 Dietz, W.B., "The Nebula Standard Computer Architecture", proc, 2nd AFSC Standardization Conference, ASD(END)-TR-82-5031, November 1982, pp 187-205.

17 Szewerenko, L., Dietz, W.B., and Ward, F.E., "Nebula: A New Architecture and its Relations to Computer Hardware", Computer Vol. 14, No.2, February 1981, pp 35-41.

18 "Nebula Instruction Set Architecture", MIL-STD-1862B, Naval Publication Center Philadelphia, PA, 3 January 1983.

19 General Electric Company, IBM, Planning Research Corp., and The Georgia Institute of Technology, "Navy Embedded Computer Accreditation Study", May 1980.

20 Clapp, J., Loebenstein, E., and Rhymer, P. "A Cost/Benefit Analysis of HOL Standardization", Mitre Report, Report No. M78-206, September 1977.

APPENDIX I: INSTRUCTION SET ARCHITECTURE PANEL FINAL REPORT EXECUTIVE SUMMARY

In November 1978, the Undersecretary of Defense, Research and Engineering asked that a panel with DoD component representation examine the question of proliferation of instruction set architectures (ISAs) in DoD systems. A panel was formed, studied the question, sought the advice of industry, and recommends standardization on a limited number of instruction set architectures be adopted as DoD policy. (A precise definition of ISA is in Attachment I.) This document is the final report of the panel.

The cost of software development and maintenance is enormous. As the microelectronic technology continues its rapid addvance, the range of applications and the number of embedded computers will dramatically increase, driving the cost of software development to new heights. In addition to excessive cost, the quality and usability of newly developed software is often poor. The software developers who create and maintain software represent a critical resource for which there is increasing competition. It has been estimated that the United States will experience a 50% shortage of software development personnel by 1985.

One factor contributing to the cost of software was the extensive use of assembly language and the proliferation of high order languages. The DoD moved to encourage the use of a few selected high order languages (DoDD 5000.31) and sponsored the development of a new high order language, Ada, to meet the documented needs of the components. The development of a sophisticated Ada Program Support Environment should provide the DoD with a powerful cost reduction opportunity. These advantages are expected to accrue over the next three to five years.

A complementary factor contributing to software cost is the number of different ISAs employed today in embedded systems. Each ISA requires unique support software and specialized training of development and maintenance programmers. In 1977, DoD Instruction 5000.xx was first proposed to limit the number of different ISAs for use in embedded sytems. The proposed instruction was widely circulated and received mixed comment from industry and DoD reviewers.

The Instruction Set Architecture Panel was chartered (Attachment II) to "recommend a course of action to eliminate the uncontrolled proliferation of embedded instruction set architectures without denying the advantages offered by developing technology." The Panel met approximately twice per month from April 1979 through January 1980. In addition regular panel members, various DoD representatives were invited to present specific viewpoints. The panel carried on many activities to ensure that industry views were received and considered. A Commerce Business Daily Bulletin notice invited industry representatives to a 2 Nov 1979 briefing on the proposed panel recommendations. Based on all of the above, the panel recommends:

that DoD issue an instruction (Attachment I) limiting the number of ISAs to be used in embedded systems:

that each DoD component develop a life cycle cost model for use in the application of the recommended instruction:

and that a separate joint program office be established to acquire or develop a large word size/large virtual address space ISA for embedded computer system.

This strategy will potentially reduce the overall cost of software for embedded computer systems by limiting the number of ISAs. Specifically, it is anticipated that the following advantages may be realized.

The use of existing software, both for support and applications, will be more feasible. The Navy experience with the AN/AYK-14 and AN/UYK-20 indicates that ISA standardization can eliminate the need to develop a great deal of support software such as compilers, operating systems, and debugging aids. The navy was able to utilize the support software developed for the AN/UYK-20 for software development for the AN/AYK-14, as the ISAs for the two machines were designed to be the same.

Run-time software can be moved from one implementation of an ISA that is inoperable (failure or battle casualty) to another still operating implementation of the same ISA.

Retraining of programmers from one ISA to another is costly and a learning curve is experienced. Much of this delay and cost can be avoided by limiting the number of ISAs in the DoD inventory.

The time required to develop software for embedded systems may be reduced.

The panel recognizes that the instruction could retard ISA improvement or innovation in DoD systems if blindly applied. The Assistant for Defense Systems Computer Resources and Electronics in the Office of the Undersecretary of Defense, Research and Engineering, with the assistance of the Management Steering Committee/Embedded Computer Resources, MSC/ECR, should monitor the implementation of the Policy by the DoD components.

This reports clarifies the provisions of the recommended instruction and should be useful in developing implementation plans consistent with the intent of the strategy.

APPENDIX II: Glossary

AIE	Ada Integrated Environment (Air Force)
AJPO	Ada Joint Program Office
ALS	Ada Language System (Army)
ANSI	American National Standards Institute
A_o	Operational Availability
APE	Advance Production Equipment
APSE	Ada Programming Support Environment
ARPANET	Advanced Research Project Agency Network
ATE	Automatic Test Equipment
ATR	Air Transport Rack
AVO	Ada Validation Office
BIT	Built In Test
C^3I	Command Control, Communications and Intelligence
CAD-CAM	Computer Aided Design-Computer Aided Manufacturing
CCP	Computer Control Panel
CMOS	Complementary MOS
CML	Current Mode Logic
CPU	Central Processor Unit
DAIS	Digital Avionics Information System
DARPA	Defense Advance Research Project Agency
DAS3	Decentralized Automated Service Support System
DCA	Defense Communication Agency
DE	Direct Execution
DESC	Defense Electronic Supply Center
DMA	Direct Memory Access
DoD	Department of Defense
DUSD(R&AT)	Deputy Under-Secretary of Defense for Research and Advanced Technology
ECL	Emitter-Coupled Logic
ECM	Electronic Counter Measure
ECP	Engineering Change Proposal
ECSPO	Embedded Computer Standardization Program Office
EEC	European Economic Community
EMI	Electromagnetic Interference
EMP	Electromagnetic Pulse
EPROM	Erasable PROM
EW	Electronic Warfare

F^3	Form Fit Function
FBM	Fleet Ballistic Missile
FSD	Full-Scale Development
GPC	General Purpose Computer
HOL	Higher Order Language
HOLWG	High Order Language Working Group
IC	Integrated Circuit
ICBM	Intercontinental Ballistic Missile
IEEE	Institute of Electrical and Electronics Engineering
I^2L	Integrated Injection Logic
I^3L	Isoplanar Integrated Injection Logic
ILS	Integrated Logistics Support
I/O	Input/Output
IOC	Input-Output Controllers
ISA	Instruction Set Architecture
KAPSE	Kernal Ada Programming Support Environment
KIT	KAPSE Interface Team
LCC	Life-Cycle Cost
LPS	Low-Power Schottky
LRM	Language Reference Manual
LRU	Line Replaceable Unit
LSA	Logistic Support Analysis
LSAR	Logistic Support Analysis Records
LSI	Large Scale Integration
MAC	Military Airlift Command
MAPSE	Minimal Ada Programming Support Environments
MCCISWG	Military Command and Control Information System Working Group
MCF	Military Computer Family
MIS	Management Information Systems
MLDT	Mean Logistic Down Time
MOS	Metal-Oxide Semiconductors
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistors
MRC	Militarized Reconfigurable Computer (Navy)
MRP	Militarized Reconfigurable Processor (Navy)
MSI	Medium-Scale Integration
MTASS	Machine Transportable Support Software
MTBF	Mean Time Between Failures
MTTR	Mean Time to Repair
MWO	Modification Work Order

NATO	North Atlantic Treaty Organization
NAVAIR	Naval Air Systems Command
NECS	Navy Embedded Computer System
NET	New Equipment Training
NMOS	N-Channel MOS
NSA	National Security Agency
nsec	Nanosecond (One billionth of a second)
NTDS	Naval Tactical Data System
ORF	Operational Readiness Float
OSD	Office of the Seretary of Defense
OUSDR&E	Office of the Under Secretary Defense Research and Engineering
P^3I	Pre-Planned Product Improvements
PDSS	Post Deployment Software Support
PROM	Programmable ROM
PSE	Programming Support Environment
QPL	Qualified Products List
QQPRI	Qualitative and Quantitative Personnel Requirements Information
RAM	Random Access Memory
RISC	Reduced Instruction Set Computer
ROM	Read Only Memory
RPSTL	Repair Parts and Special Tools List
SAM	Surface to Air Missile
SCADC	Standard Central Air Data Computer
SCD	Specification Control Drawings
SEAFAC	Systems Engineering Avionics Facility
SEE	Software Engineering Environment
SNAP	Shipboard Non-Tactical Automated Data Processing
SON	Statement of Operational Readiness
SOS	Silicon-On-Sapphire
SRA	Shop Replaceable Assemblies
SSI	Small-Scale Integration
STARS	Software Technology for Adaptable, Reliable Systems
TM/MAC	Technical Manual/Maintenance Allocation Charts
TMDE	Test, Measurement, Diagnostic/Support Equipment
TTL	Transistor-Transistor Logic
uP	Microprocessor
UUT	Unit Under Test
VHSIC	Very High Speed Integrated Circuit
VLSIC	Very Large Scale Integrated Circuit